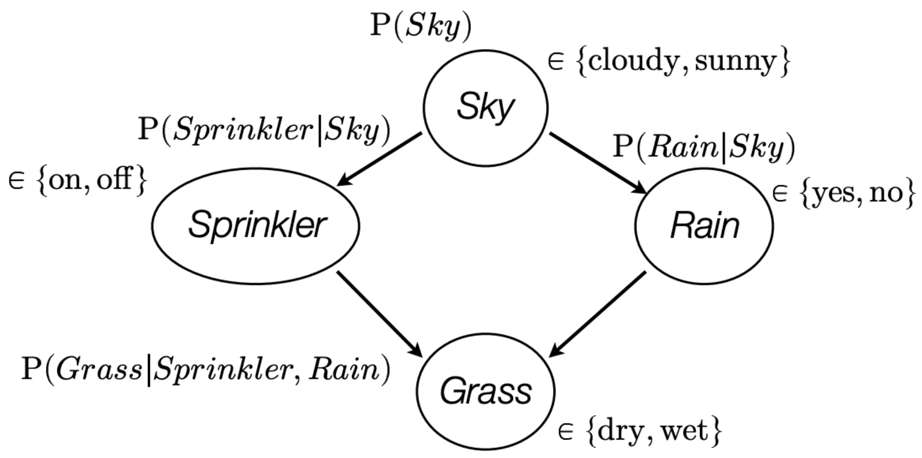
Probabilistic Inference

In this experiment, I performed exact and approximated inference on a simple Bayesian Network.

PART I: Defining the Network

I used the following network. The posterior probabilities indicated by the arrows on the graph are given by the tables below. I put the function to return an entry of the table below each table. In the code I used binary digits to represent the binary values of the four variables.


$\mathrm{P}(Sky)$
Sky
$\in \{\mathrm{cloudy}, \mathrm{sunny}\}$
$\mathrm{P}(Sprinkler|Sky)$
$\mathrm{P}(Rain|Sky)$
$\in \{\mathrm{on}, \mathrm{off}\}$
Sprinkler
Rain
$\in \{\mathrm{yes}, \mathrm{no}\}$
$\mathrm{P}(Grass|Sprinkler, Rain)$
Grass
$\in \{\mathrm{dry}, \mathrm{wet}\}$

a. P(sky)

Sky	P(sky)
Cloudy	0.5
Sunny	0.5

b. P(rain|sky)

Sky	P(rain=false|sky)	P(rain=true|sky)
Cloudy	0.3	0.7
Sunny	0.99	0.01

```
function p = p_rain_given_sky(rain, sky)
        % Returns P(rain|sky) with given rain and sky values
        % rain = 0(false) or 1(true)
        % sky = 0(cloudy) or 1(sunny)

        % Sky = cloudy, sky = sunny
        table_rain_given_sky = [[0.3, 0.99],                % Rain = false
                                [0.7, 0.01]];        % Rain = true
        p = table_rain_given_sky(rain + 1, sky + 1);
end
```

c. P(sprinkler|sky)

Sky	P(sprinkler=off|sky)	P(sprinkler=on|sky)
Cloudy	0.5	0.5
Sunny	0.1	0.9

```
function p = p_spr_given_sky(spr, sky)
        % Returns P(spr|sky) with given spr and sky values
        % spr = 0(off) or 1(on)
        % sky = 0(cloudy) or 1(sunny)

        % Sky = cloudy, sky = sunny
        table_spr_given_sky = [[0.5, 0.1],                % Sprinkler = off
                               [0.5, 0.9]];     % Sprinkler = on
        p = table_spr_given_sky(spr + 1, sky + 1);
end
```

d. P(grass|rain, sprinkler)

Rain	Sprinkler	P(grass=dry|rain,sprinkler)	P(grass=wet|rain,sprinkler)
False	Off	0.99	0.01
False	On	0.1	0.9
True	Off	0.1	0.9
True	On	0.01	0.99

```
function p = p_grass_given_rain_spr(grass, rain_spr)
        % Returns P(grass|rain, spr) with given grass, rain, spr values
        % spr = 0(off) or 1(on)
        % rain = 0(false) or 1(true)
        % grass = 0(dry) or 1(wet)

        % rain_spr = 00 or 01 or 10 or 11
        % Ex: rain = false, spr = on
        % column index = bin01 + 1 = dec2
        table_grass_given_rain_spr = [[0.99, 0.1, 0.1, 0.01],      % Grass = dry
                                      [0.01, 0.9, 0.9, 0.99]];     % Grass = wet
        p = table_grass_given_rain_spr(grass + 1, rain_spr + 1);
end
```

PART II: Inference

a. Combinations

Here I calculated the probability of all combinations based on the graphical model. The following function and script generates the table of probabilities and adds up all of them.

$$P(sky, sprinkler, rain, grass)$$
$$= P(sky)P(sprinkler|sky)P(rain|sky)P(grass|sprinkler, rain)$$

```
function p = p_combi(sky, spr, rain, grass)
% Return the probability of any combinations of variables
% Using bayesian techniques on graphical models
% P(sky, spr, rain, grass) = P(sky) * P(rain|sky) * P(spr|sky) * P(grass|rain, spr)
% sky = 0(cloudy) or 1(sunny)
% spr = 0(off) or 1(on)
% rain = 0(false) or 1(true)
% grass = 0(dry) or 1(wet)
% rain_spr = 00 or 01 or 10 or 11
        rain_spr = rain * 2 + spr;
        p_sky = 0.5;
        p1 = p_rain_given_sky(rain, sky);
        p2 = p_spr_given_sky(spr, sky);
        p3 = p_grass_given_rain_spr(grass, rain_spr);
        p = p_sky * p1 * p2 * p3;
end

% Generate probability table of all combinations
% Sky, sprinkler, rain, grass = 0 or 1
% Ex: sky = sunny, spr = off, rain = false, grass = wet
%     index = bin1001 + 1 = dec10
% 2^4 = 16 combinations

prob = zeros(1, 16);
sum_all = 0;
for sky = 0 : 1
    for spr = 0 : 1
        for rain = 0 : 1
            for grass = 0 : 1
                index = sky * 8 + spr * 4 + rain * 2 + grass + 1;
                prob(1, index) = p_combi(sky, spr, rain, grass);
                sum_all = sum_all + prob(1, index);
            end
        end
    end
end

clear sky spr rain grass index;

combi = {'0 0 0 0', '0 0 0 1', '0 0 1 0', '0 0 1 1', '0 1 0 0', '0 1 0 1', '0 1 1 0',
'0 1 1 1', ...
```

```
    '1 0 0 0', '1 0 0 1', '1 0 1 0', '1 0 1 1', '1 1 0 0', '1 1 0 1', '1 1 1 0', '1 1
1 1'};
varNames = {'Sky_Sprinkler_Rain_Grass', 'Probability'};
T_combi = table(combi.', prob.', 'VariableNames', varNames);

clear combi varNames;
```

sum_all indeed turns out to be 1. And the table generated is shown below. We can see that the values are quite reasonable. For instance, when the sprinkler is on and it's raining, it's very unlikely for the grass not to be wet.

```
Sky_Sprinkler_Rain_Grass    Probability
________________________    ___________

       '0 0 0 0'              0.07425
       '0 0 0 1'              0.00075
       '0 0 1 0'               0.0175
       '0 0 1 1'               0.1575
       '0 1 0 0'               0.0075
       '0 1 0 1'               0.0675
       '0 1 1 0'              0.00175
       '0 1 1 1'              0.17325
       '1 0 0 0'             0.049005
       '1 0 0 1'             0.000495
       '1 0 1 0'                5e-05
       '1 0 1 1'              0.00045
       '1 1 0 0'              0.04455
       '1 1 0 1'              0.40095
       '1 1 1 0'              4.5e-05
       '1 1 1 1'             0.004455
```

b. Exact Inference

Now we want to calculate the joint posterior over all unobserved variables conditioned on the grass being dry, and the grass being wet. I used the following formula to calculate.

$$P(sky, sprinkler, rain | grass = dry) = \frac{P(sky, sprinkler, rain, grass = dry)}{P(grass = dry)}$$

$$= \frac{P(sky, sprinkler, rain, grass = dry)}{\sum_{sky}\sum_{sprinkler}\sum_{rain} P(sky, sprinkler, rain, grass = dry)}$$

The following function and script perform the exact inference and yield 2 tables, with respect to the two values of grass.

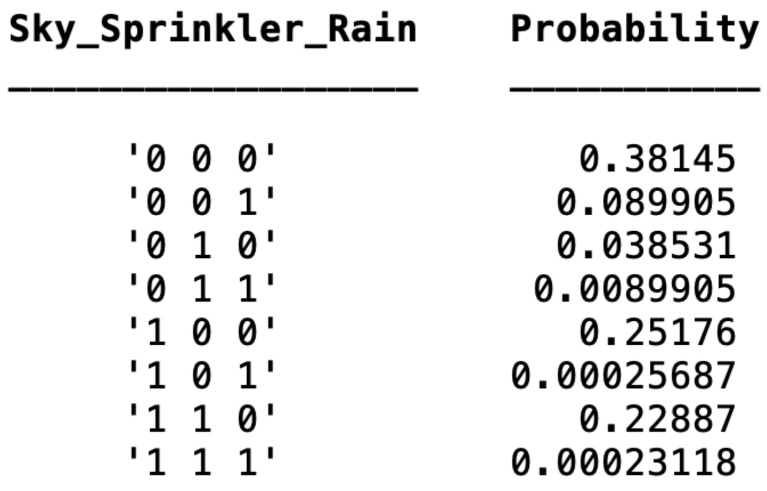
```
function p = p_given_grass(sky, spr, rain, grass)
```

```
    % Returns P(sky, spr, rain|grass)
    % sky = 0(cloudy) or 1(sunny)
    % spr = 0(off) or 1(on)
    % rain = 0(false) or 1(true)
    % grass = 0(dry) or 1(wet)

        % P(sky, spr, rain, grass)
        p_all = p_combi(sky, spr, rain, grass);

        % P(grass) = Sum(sky, spr, rain) of P(grass, sky, spr, rain)
        p_grass = 0;
        for sky = 0 : 1
                for spr = 0 : 1
                        for rain = 0 : 1
                                p_grass = p_grass + p_combi(sky, spr, rain, grass);
                        end
                end
        end

        % P(sky, spr, rain|grass)
        % = P(sky, spr, rain, grass) / P(grass)
        p = p_all / p_grass;
end

% By exact inference
% Generate two tables of probabilities of all combinations of sky, sprinkler, rain
% Given grass is dry and grass is wet
% Index is given by sky, spr, rain
% Ex: sky = sunny, spr = on, rain = false
%     index = bin110 + 1 = dec7

prob_dryEx = zeros(1, 8);
prob_wetEx = zeros(1, 8);
sum_dryEx = 0;
sum_wetEx = 0;

% Calculate the probability
% P(sky, sprinkler, rain|grass=dry) and  P(sky, sprinkler, rain|grass=wet)
for sky = 0 : 1
        for spr = 0 : 1
                for rain = 0 : 1
                        index = sky * 4 + spr * 2 + rain + 1;
                        prob_dryEx(1, index) = p_given_grass(sky, spr, rain, 0);
                        prob_wetEx(1, index) = p_given_grass(sky, spr, rain, 1);
                        sum_dryEx = sum_dryEx + prob_dryEx(1, index);
                        sum_wetEx = sum_wetEx + prob_wetEx(1, index);
                end
        end
end
```

```
clear sky spr rain index;

varNames = {'Sky_Sprinkler_Rain', 'Probability'};
combi_dry = {'0 0 0', '0 0 1', '0 1 0', '0 1 1', '1 0 0', '1 0 1', '1 1 0', '1 1 1'};
T_dryEx = table(combi_dry.', prob_dryEx.', 'VariableNames', varNames);
combi_wet = {'0 0 0', '0 0 1', '0 1 0', '0 1 1', '1 0 0', '1 0 1', '1 1 0', '1 1 1'};
T_wetEx = table(combi_wet.', prob_wetEx.', 'VariableNames', varNames);

clear combi_dry combi_wet varNames;
```

The sum_dryEx and sum_wetEx yielded both turn out to be 1. The followings are the 2 tables yielded.

GRASS = DRY



```
Sky_Sprinkler_Rain     Probability
__________________     ___________

     '0  0  0'            0.38145
     '0  0  1'           0.089905
     '0  1  0'           0.038531
     '0  1  1'          0.0089905
     '1  0  0'            0.25176
     '1  0  1'         0.00025687
     '1  1  0'            0.22887
     '1  1  1'         0.00023118
```

GRASS = WET

```
Sky_Sprinkler_Rain    Probability
__________________    ___________

     '0 0 0'          0.00093127
     '0 0 1'             0.19557
     '0 1 0'            0.083814
     '0 1 1'             0.21512
     '1 0 0'          0.00061464
     '1 0 1'          0.00055876
     '1 1 0'             0.49786
     '1 1 1'           0.0055318
```

c. Gibbs Sampling

In this part I performed approximated inference using Gibbs Sampling to approximate the same joint posteriors as in b. For each value of grass, I randomly assigned values to the other three variables and then update their values one by one in each round then make them a sample. I generated 1000 samples for each value of grass. The formula to update variable sky is as follows. The ones to update sprinkler and rain are similar.

$$P(sky|sprinkler, rain, grass) = \frac{P(sky, sprinkler, rain, grass)}{p(sprinkler, rain, grass)}$$

$$= \frac{P(sky, sprinkler, rain, grass)}{P(sky = cloudy|sprinkler, rain, grass) + P(sky = sunny|sprinkler, rain, grass)}$$

Now I use the following function and script to generate the probability tables as in b. The function returns 2 1000 x 3 samples and the script uses that to calculate the probabilities.

```
function [spl_dry, spl_wet] = genGibbsSamples(prob_all)
    % porb_all is the table of probabilities of all combinations of P(sky,
    % spr, rain, grass)s
    % Performs Gibbs sampling on the graphical model
    % Returns 2 * 1000 samples given grass = dry and grass = wet, respectively
    % Column 1 = sky, column 2 = sprinkler, column 3 = rain

    % Grass = dry
    curr = [binornd(1, 0.5), binornd(1, 0.5), binornd(1, 0.5)];     % Current values
    spl_dry = zeros(1000, 3);
    for t = 1 : 1000
        % Update current values
        % P(sky|spr, rain, grass) = P(sky, spr, rain, grass) / P(spr, rain, grass)
        index = curr(1, 2) * 4 + curr(1, 3) * 2 + 1;
        p_sky = prob_all(1, index + 8) / (prob_all(1, index + 8) + prob_all(1,
index));
        curr(1, 1) = binornd(1, p_sky);
        % P(spr|sky, rain, grass) = P(sky, spr, rain, grass) / P(sky, rain, grass)
        index = curr(1, 1) * 8 + curr(1, 3) * 2 + 1;
        p_spr = prob_all(1, index + 4) / (prob_all(1, index + 4) + prob_all(1,
index));
        curr(1, 2) = binornd(1, p_spr);
        % P(rain|sky, spr, grass) = P(sky, spr, rain, grass) / P(sky, spr, grass)
        index = curr(1, 1) * 8 + curr(1, 2) * 4 + 1;
        p_rain = prob_all(1, index + 2) / (prob_all(1, index + 2) + prob_all(1,
index));
        curr(1, 3) = binornd(1, p_rain);

        % Copy the values into samples matrix
        spl_dry(t, 1) = curr(1, 1);
        spl_dry(t, 2) = curr(1, 2);
        spl_dry(t, 3) = curr(1, 3);
    end

    % Grass = wet
    curr = [binornd(1, 0.5), binornd(1, 0.5), binornd(1, 0.5)];     % Current values
    spl_wet = zeros(1000, 3);
    for t = 1 : 1000
        % Update current values
        % P(sky|spr, rain, grass) = P(sky, spr, rain, grass) / P(spr, rain, grass)
        index = curr(1, 2) * 4 + curr(1, 3) * 2 + 1 + 1;
        p_sky = prob_all(1, index + 8) / (prob_all(1, index + 8) + prob_all(1,
index));
        curr(1, 1) = binornd(1, p_sky);
        % P(spr|sky, rain, grass) = P(sky, spr, rain, grass) / P(sky, rain, grass)
        index = curr(1, 1) * 8 + curr(1, 3) * 2 + 1 + 1;
        p_spr = prob_all(1, index + 4) / (prob_all(1, index + 4) + prob_all(1,
index));
        curr(1, 2) = binornd(1, p_spr);
        % P(rain|sky, spr, grass) = P(sky, spr, rain, grass) / P(sky, spr, grass)
        index = curr(1, 1) * 8 + curr(1, 2) * 4 + 1 + 1;
        p_rain = prob_all(1, index + 2) / (prob_all(1, index + 2) + prob_all(1,
index));
        curr(1, 3) = binornd(1, p_rain);

        % Copy the values into samples matrix
        spl_wet(t, 1) = curr(1, 1);
        spl_wet(t, 2) = curr(1, 2);
        spl_wet(t, 3) = curr(1, 3);
    end

end



% By Gibbs sampling
% Generate two tables of probabilities of all combinations of sky, sprinkler, rain
% Given grass is dry and grass is wet
% Index is given by sky, spr, rain
% Ex: sky = sunny, spr = on, rain = false
%     index = bin110 + 1 = dec7

enumCombi;
clear sum_all T_combi;
[spl_dry, spl_wet] = genGibbsSamples(prob);     % Get samples

cnt_dry = zeros(1, 8);
cnt_wet = zeros(1, 8);
prob_dryAp = zeros(1, 8);
prob_wetAp = zeros(1, 8);
sum_dryAp = 0;
sum_wetAp = 0;

% Loop over each entry of the samples
% Count number of appearance of each 3-tuple combinations
for t = 1 : 1000
    index = spl_dry(t, 1) * 4 + spl_dry(t, 2) * 2 + spl_dry(t, 3) + 1;
    cnt_dry(1, index) = cnt_dry(1, index) + 1;
    index = spl_wet(t, 1) * 4 + spl_wet(t, 2) * 2 + spl_wet(t, 3) + 1;
    cnt_wet(1, index) = cnt_wet(1, index) + 1;
end

% Calculate the probability
% P(sky, sprinkler, rain|grass=dry) and  P(sky, sprinkler, rain|grass=wet)
for i = 1 : 8
    prob_dryAp(1, i) = cnt_dry(1, i) / 1000;
    sum_dryAp = sum_dryAp + prob_dryAp(1, i);
    prob_wetAp(1, i) = cnt_wet(1, i) / 1000;
    sum_wetAp = sum_wetAp + prob_wetAp(1, i);
end


clear i t prob index cnt_dry cnt_wet;

varNames = {'Sky_Sprinkler_Rain', 'Probability'};
combi_dry = {'0 0 0', '0 0 1', '0 1 0', '0 1 1', '1 0 0', '1 0 1', '1 1 0', '1 1 1'};
T_dryAp = table(combi_dry.', prob_dryAp.', 'VariableNames', varNames);
combi_wet = {'0 0 0', '0 0 1', '0 1 0', '0 1 1', '1 0 0', '1 0 1', '1 1 0', '1 1 1'};
T_wetAp = table(combi_wet.', prob_wetAp.', 'VariableNames', varNames);

clear combi_dry combi_wet varNames;
```

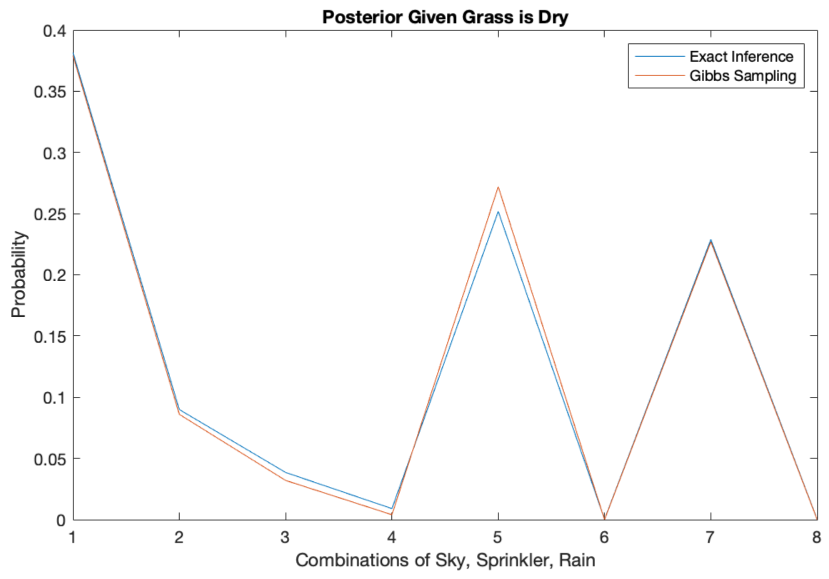
sum_dryAp, sum_wetAp both turn out to be 1, which confirms the correctness. The 2 tables are as follows. From intuition, the tables seem correct, despite the 0s that appear on them. Comparing with the tables we got from exact inference, there are clearly some errors from only 1000 samples when we use sampling. Nevertheless, when I compare them on the same plot the probabilities from the two methods fit very well.

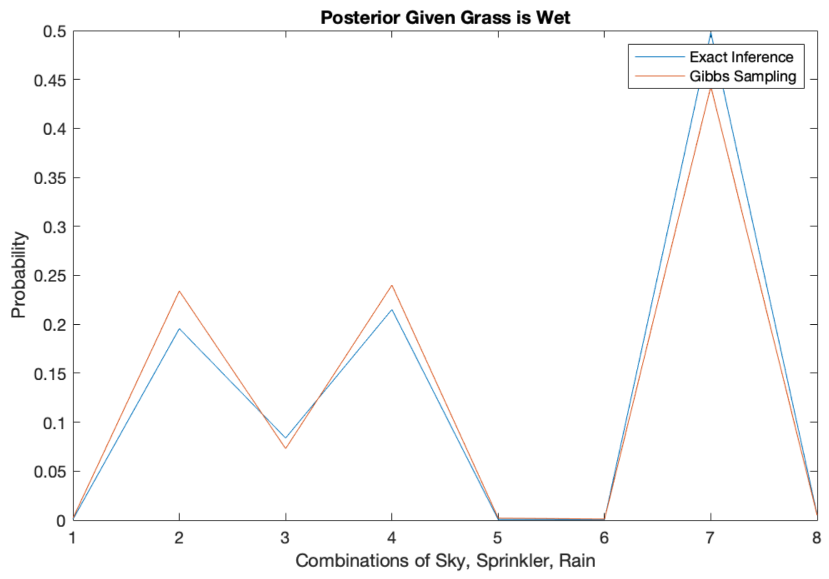
GRASS = DRY

```
Sky_Sprinkler_Rain    Probability
__________________    ___________

     '0 0 0'             0.403
     '0 0 1'             0.089
     '0 1 0'              0.04
     '0 1 1'             0.007
     '1 0 0'             0.238
     '1 0 1'                 0
     '1 1 0'             0.222
     '1 1 1'             0.001
```

GRASS = WET

```
Sky_Sprinkler_Rain    Probability
__________________    ___________

     '0 0 0'                 0
     '0 0 1'             0.208
     '0 1 0'             0.079
     '0 1 1'              0.23
     '1 0 0'                 0
     '1 0 1'                 0
     '1 1 0'             0.475
     '1 1 1'             0.008
```





d. Pairwise Correlation

I used the following script to compute the pairwise correlation between variables in the 2x1000 samples I got from Gibbs Sampling.

```
pCondAppro;

clear prob_dryAp prob_wetAp sum_dryAp sum_wetAp T_dryAp T_wetAp;

corr_dry = corrcoef(spl_dry);
corr_wet = corrcoef(spl_wet);

varNames = {'Correlation', 'Sky', 'Sprinkler', 'Rain'};
comp = {'Sky', 'Sprinkler', 'Rain'};
```

```
T_corr_dry = table(comp.', corr_dry(1,:).', corr_dry(2,:).', corr_dry(3,:).',
'VariableNames', varNames);
T_corr_wet = table(comp.', corr_wet(1,:).', corr_wet(2,:).', corr_wet(3,:).',
'VariableNames', varNames);

clear comp varNames;
```

Now we can get two 3x3 correlation tables as shown below. The signs on the entries are the same in the two tables, which is as expected. The diagonals are 1, also as expected because the correlation between a variable and itself is 1. We can see that there are some negative correlations. For example, ‘sky’ and ‘rain’ have a negative correlation. This is intuitively correct because when sky=1(sunny), it’s very unlikely for the rain to be 1(true). Also, the correlation between ‘Sky’ and ‘sprinkler’ is positive because when it’s sunny the sprinkler is likely to be on.

GRASS = DRY

```
Correlation      Sky      Sprinkler      Rain
___________    ________   _________    ________

'Sky'                 1     0.46697    -0.33439
'Sprinkler'     0.46697           1    -0.15778
'Rain'         -0.33439    -0.15778           1
```

GRASS = WET

```
Correlation      Sky      Sprinkler      Rain
___________    ________   _________    ________

'Sky'                 1     0.50901    -0.82058
'Sprinkler'     0.50901           1     -0.5994
'Rain'         -0.82058     -0.5994           1
```

PART III: Repeat Inference on Deterministic Probabilities

In this section I changed the probabilities to make them more deterministic (closer to 1-0 instead of 0.5-0.5) and repeated the exact and approximated inference on them.

a. New Probabilities

The new probabilities are shown in tables below.

Sky	P(sky)
Cloudy	0.2
Sunny	0.8

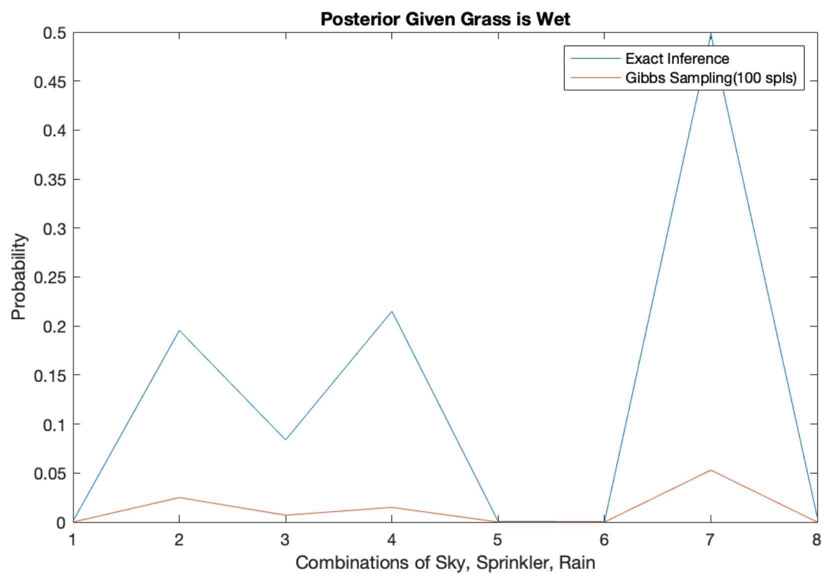
Sky	P(rain=false|sky)	P(rain=true|sky)
Cloudy	0.3	0.7
Sunny	0.99	0.01

Sky	P(sprinkler=off|sky)	P(sprinkler=on|sky)
Cloudy	0.8	0.2
Sunny	0.1	0.9

Rain	Sprinkler	P(grass=dry|rain,sprinkler)	P(grass=wet|rain,sprinkler)
False	Off	0.99	0.01
False	On	0.1	0.9
True	Off	0.1	0.9
True	On	0.01	0.99

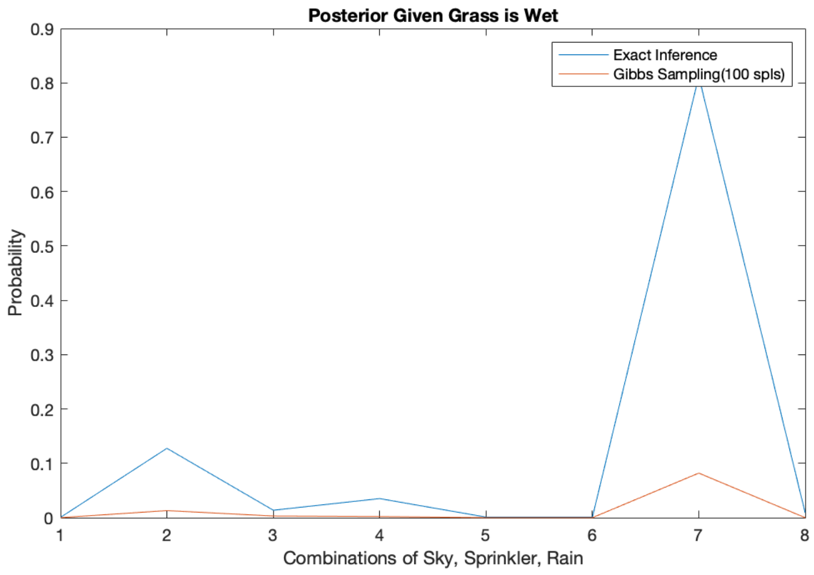
b. Inference Comparison

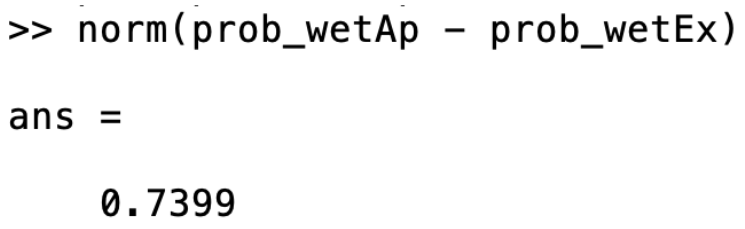
I repeated the process of calculating the exact and approximated inferences. For Gibbs Sampling I only generated 100 samples for better illustration of the difference. The first plot is comparing the probabilities when using the original probabilities given that grass is wet. I also calculated the norm of the differences.



```
>> norm(prob_wetAp - prob_wetEx)

ans =

    0.5225
```

Next, I repeated using the new probability tables with more deterministic probabilities. Here's the plot and norm of the differences.





```
>> norm(prob_wetAp - prob_wetEx)

ans =

    0.7399
```

The difference may not be clear from the plot, but the norm of differences increased from about 0.5 to about 0.7. It means that the error definitely increases when the conditional probabilities become more deterministic. The reason is probably as follows. When the probability becomes more deterministic, the chance of a sample drawn from the distribution to reach the less probable values decreases significantly, so we might only get very few samples from that value. However, we need those samples to calculate the probability in Gibbs Sampling. It’s very likely to cause a lot of error especially when all the conditional probabilities are deterministic, and that we only have 100 samples. That’s why when I repeated using 1000 samples, the error difference is not that much.