1. You are indifferent between receiving A for sure and a lottery that gives you B with a probability of 0.9 and C with a probability of 0.1. You are also indifferent between receiving A for sure and a lottery that gives you B with a probability of 0.7 and D with a probability of 0.3. Your preferences satisfy the von N-M axioms: If the utility of B = 1 and your utility for C is 0, then what are your utilities for A and D?

$$A = 1$$

$$u_A A \sim u_B B + u_C C \sim 1(.9) + 0(0.1)$$

$$u_A \sim \boxed{0.9}$$

$$u_A A \sim u_B B + u_D D$$

$$u_D \sim \frac{u_A A - u_B B}{D} \sim \frac{0.9 - (1)(0.7)}{0.3}$$

$$u_D \sim \boxed{0.6667}$$

2. For the Motor Design problem, use the Monte Carlo method to compare the expected utility of two different motor designs. Use the following uncertain parameters: Use the exponential forms in the slides and assume a risk aversion coefficient of 10, a "high" value of 40 and a "low" value of 2. In this problem, assume you prefer less weight. Which design has higher expected utility?

$$D\ N(7.5, 0.5)$$

$$L\ N(9.5, 0.5)$$

$$\rho_{cu}\ N(8.94e3, 100)$$

$$\rho_{fe}\ N(7.98e3, 100)$$

Using 10000 samples I found that Design 2 had a higher utility (See attached code).

Design 1

- Expected Utility : $\boxed{0.9203}$
- Expected Weight: $\boxed{16.77kg}$

Design 2

- Expected Utility : $\boxed{0.9798}$
- Expected Weight: $\boxed{8.18kg}$

3. Repeat problem 2 but use the FFNI (i.e. Gaussian quadrature) method using 3 nodes: Which design has higher expected utility? If you were to compare motor designs 1 and 2 as a risk neutral decision maker, which one would be preferred?

$$D\ N(6.5, 0.5)$$
$$L\ N(11.5, 0.5)$$
$$\rho_{cu}\ N(8.94e3, 100)$$
$$\rho_{fe}\ N(7.98e3, 100)$$

With a risk aversion coefficient of 10 I found that design 2 had a higher utility (See attached code).

Design 1

- Expected Weight: $\boxed{15.67kg}$
- CE: $\boxed{16.21kg}$
- Expected Utility : $\boxed{0.9310}$

Design 2

- Expected Weight: $\boxed{9.06kg}$
- CE: $\boxed{9.31kg}$
- Expected Utility : $\boxed{0.9758}$

As a risk neutral decision maker I found that design 2 still had a higher utility (See attached code).

Design 1

- Expected Weight: $\boxed{15.67kg}$
- CE: $\boxed{15.74kg}$
- Expected Utility : $\boxed{0.6403}$

Design 2

- Expected Weight: $\boxed{9.06kg}$
- CE: $\boxed{9.00kg}$
- Expected Utility : $\boxed{0.8142}$

4. Redo problem 3 but use the Robust Design formulation in which $CE = \mu + \frac{\sigma^2}{2\rho}$ and the FTNI method (use 5 nodes): Compare the CE equivalents from both problems 3 and 4—how do they differ?

With a risk aversion coefficient of 10 I found that design 2 had a higher utility (See attached code).

Design 1

- Expected Weight: $\boxed{15.67kg}$
- CE: $\boxed{15.90kg}$
- Expected Utility : $\boxed{0.9310}$

Design 2

- Expected Weight: $\boxed{9.06kg}$
- CE: $\boxed{9.22kg}$
- Expected Utility : $\boxed{0.9758}$

As a risk neutral decision maker I found that design 2 still had a higher utility (See attached code).

Design 1

- Expected Weight: $\boxed{15.67kg}$
- CE: $\boxed{15.66kg}$
- Expected Utility : $\boxed{0.6403}$

Design 2

- Expected Weight: $\boxed{9.06kg}$
- CE: $\boxed{9.06kg}$
- Expected Utility : $\boxed{0.8142}$

For design 1 I found that using the Robust Design formulation resulted in a slightly lower CE than when using the exponential form for both risk adverse (15.09kg $<$ 16.21kg) and risk neutral view points (15.66kg $<$ 15.74kg). For design 2 I found that using the Robust Design formulation resulted in a slightly lower CE for risk adverse (9.22 kg $<$ 9.31kg) but a slightly higher CE for risk neutral (9.06 kg $>$ 9.00 kg).(See attached code)

```
% Design Under Uncertainty : HW4 Utility Theory
% Heather Miller
% Started 5/25/20
% Due 5/27/20

% NOTE: I have modified the provided motorDesignHW4.m file to take in
 only
% variables and design number (1 or 2)

clear all
clc

%adding CEtools folder to paths
o_path = path;
path(o_path, 'C:\Users\MTH\Documents\Gradwork
\Spring2020\Design_Under_Uncertainty\HW\HW4\CEtools');

problem = 4;

% used for problems 2-4 remaining problems
samples = 10000;
rho = inf; %10 for risk adverse inf for risk neutral
high = 40;
low = 2;

% Problem 1 - Determining the utility of A and D
% uA~uB1+uC and uA~uB2+uD
if problem == 1
    prob_b_1 = 0.9;
    prob_b_2 = 0.7;
    prob_c = 0.1;
    prob_d = 0.3;

    utility_b = 1;
    utility_c = 0;

    utility_a = prob_b_1*utility_b + prob_c*utility_c;
    utility_d = (utility_a-(prob_b_2*utility_b))/prob_d;

    fprintf('Problem 1\n')
    fprintf('Utility of A: %.4f\n', utility_a)
    fprintf('Utility of D: %.4f\n', utility_d)
end

% problem 2 - Monte Carlo method to compare the expected utility of
% two different motor models
if problem == 2
    d_1 = NaN(1, samples);
    d_2 = NaN(1, samples);
    ut_d1 = NaN(1, samples);
    ut_d2 = NaN(1, samples);
```

```
    for i = 1:samples
        %design variables
        D = normrnd(7.5, 0.5);
        L = normrnd(9.5, 0.5);
        dcu = normrnd(8.94e3, 100);
        dfe = normrnd(7.98e3, 100);

        %weights given randomly generated values
        d_1(i) = motorDesignHW4(D, L, dcu, dfe, 1);
        d_2(i) = motorDesignHW4(D, L, dcu, dfe, 2);

        %utility
        ut_d1(i) = utility(d_1(i), rho, high, low);
        ut_d2(i) = utility(d_2(i), rho, high, low);
    end

    fprintf('Problem 2\n')
    fprintf('Design 1: %.4f\n', sum(d_1)/samples)
    fprintf('Utility 1: %.4f\n', sum(ut_d1)/samples)
    fprintf('Design 2: %.4f\n', sum(d_2)/samples)
    fprintf('Utility 2: %.4f\n', sum(ut_d2)/samples)
end

% problem 3 - Using FFNI method with 3 nodes to compare the expected
% utility of 2 different motor models
if problem ==3
    indx = fullfact([3 3 3 3]);
    %design variables
    [D, d_weight] = qnwnorm(3, 6.5, 0.5^2);
    [L, l_weight] = qnwnorm(3, 11.5, 0.5^2);
    [dcu, dcu_weight] = qnwnorm(3, 8.94e3, 100^2);
    [dfe, dfe_weight] = qnwnorm(3, 7.98e3, 100^2);

    design_1 = NaN(1, length(indx));
    design_2 = NaN(1, length(indx));
    W = NaN(1, length(indx));
    ut_d1 = NaN(1, length(indx));
    ut_d2 = NaN(1, length(indx));

    d1_CE = NaN(1, length(indx));
    d2_CE = NaN(1, length(indx));

    for i = 1: length(indx)
        % the index for each variable to use on this interation
        D_idx = indx(i, 1);
        L_idx = indx(i, 2);
        dcu_idx = indx(i, 3);
        dfe_idx = indx(i, 4);
        %designs
        design_1(i) = motorDesignHW4(D(D_idx),
 L(L_idx),dcu(dcu_idx),...
            dfe(dfe_idx), 1);
        design_2(i) = motorDesignHW4(D(D_idx),
 L(L_idx),dcu(dcu_idx),...
```

```
            dfe(dfe_idx), 2);
        % weights
        W(i) =
 d_weight(D_idx)*l_weight(L_idx)*dcu_weight(dcu_idx)*....
            dfe_weight(dfe_idx);
        %utility
        ut_d1(i) =  utility(design_1(i), rho, high, low)*W(i);
        ut_d2(i) =  utility(design_2(i), rho, high, low)*W(i);

        %CE design 1
        d1_CE(i) = exp(design_1(i)/rho);

         %CE design 2
        d2_CE(i) = exp(design_2(i)/rho);


    end
    % Design 1 moments
    [d1_mean, d1_sigma, d1_skew, d1_kurt] = multi_moments(design_1,
 W);
    % Design 2 moments
    [d2_mean, d2_sigma, d2_skew, d2_kurt] = multi_moments(design_2,
 W);

    %CE design 1
    if rho == inf
        d1_CE = CE_exponential(sum(design_1)/length(indx), rho);
    else
        d1_CE = CE_exponential(sum(d1_CE)/length(indx), rho);
    end

    if rho == inf
        d2_CE = CE_exponential(sum(design_2)/length(indx), rho);
    else
        d2_CE = CE_exponential(sum(d2_CE)/length(indx), rho);
    end



    fprintf('Problem 3 : rho = %i\n', rho)
    fprintf('Design 1: %.4f\n', d1_mean)
    fprintf('CE 1: %.4f\n', d1_CE)
    fprintf('Utility 1: %.4f\n', sum(ut_d1))

    fprintf('Design 2: %.4f\n', d2_mean)
    fprintf('CE 2: %.4f\n', d2_CE)
    fprintf('Utility 2: %.4f\n', sum(ut_d2))


end

% problem 4 - using Robust Design formulation and the FTNI with 5
 nodes
% to compare the certainty equivalent of two motor models
```

```
if problem == 4
    indx = fullfact([5 5 5 5]);

    %design variables
    [D, d_weight] = qnwnorm(5, 6.5, 0.5^2);
    [L, l_weight] = qnwnorm(5, 11.5, 0.5^2);
    [dcu, dcu_weight] = qnwnorm(5, 8.94e3, 100^2);
    [dfe, dfe_weight] = qnwnorm(5, 7.98e3, 100^2);

    design_1 = NaN(1, length(indx));
    design_2 = NaN(1, length(indx));
    W = NaN(1, length(indx));
    ut_d1 = NaN(1, length(indx));
    ut_d2 = NaN(1, length(indx));

    for i = 1: length(indx)
        % the index for each variable to use on this interation
        D_idx = indx(i, 1);
        L_idx = indx(i, 2);
        dcu_idx = indx(i, 3);
        dfe_idx = indx(i, 4);
        %designs
        design_1(i) = motorDesignHW4(D(D_idx),
 L(L_idx),dcu(dcu_idx),...
            dfe(dfe_idx), 1);
        design_2(i) = motorDesignHW4(D(D_idx),
 L(L_idx),dcu(dcu_idx),...
            dfe(dfe_idx), 2);
        % weights
        W(i) =
 d_weight(D_idx)*l_weight(L_idx)*dcu_weight(dcu_idx)*....
            dfe_weight(dfe_idx);
        %utility
        ut_d1(i) =  utility(design_1(i), rho, high, low)*W(i);
        ut_d2(i) =  utility(design_2(i), rho, high, low)*W(i);


    end
    % Design 1 moments
    [d1_mean, d1_sigma, d1_skew, d1_kurt] = multi_moments(design_1,
 W);
    % Design 2 moments
    [d2_mean, d2_sigma, d2_skew, d2_kurt] = multi_moments(design_2,
 W);

    %CE design 1
    d1_CE = d1_mean + ((d1_sigma^2)/(2*rho));
    d2_CE = d2_mean + ((d2_sigma^2)/(2*rho));


    fprintf('Problem 4: rho = %i\n', rho)
    fprintf('Design 1: %.4f\n', d1_mean)
    fprintf('CE 1: %.4f\n', d1_CE)
```

```
    fprintf('Utility 1: %.4f\n', sum(ut_d1))

    fprintf('Design 2: %.4f\n', d2_mean)
    fprintf('CE 2: %.4f\n', d2_CE)
    fprintf('Utility 2: %.4f\n', sum(ut_d2))
end
```

```
function output = utility(x, rho, high, low)
    if isinf(rho)
        output = (high-x)/(high-low);
    else
        output = (exp(-(high-x)/rho)-1)/(exp(-(high-low)/rho)-1);

    end

end
```

```
function [mean, sigma, skew, kurt] = multi_moments(output, Weight)
    mean = dot(Weight, output);
    sigma = sqrt(dot(Weight, (output-mean).^2));
    skew = dot(Weight, (output-mean).^3)/sigma.^3;
    kurt = dot(Weight, (output-mean).^4)/sigma.^4;
end
```

```
function certainty_equivalence = CE_exponential(x, rho)
    if isinf(rho)
        certainty_equivalence = x;
    else
        certainty_equivalence = rho*log(x);

    end

end
```