EE525 - Lab Project 1

Authors: Grant Gallagher, Alexander Goldstein, Yu-Shun Hsiao, Jacobis, Soriano Pelaez

Due: 10/8/2021

Objective:

Table of Contents

Initialization

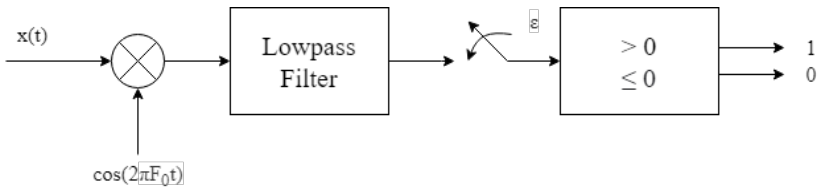
This section of code is used to clear the workspace of all variables and figures, so that the program can be run from scratch.

```
clc;    % Clear command window
clear;  % Clear all variables from workspace
clf;    % Clear all figures
```

1. Digital Communications

Problem Statement

In a phase-shift keyed (**PSK**) digital communication system, a binary digit (also termed a bit), which is either a "**0**" or a "**1**", is communicated to a receiver by sending either $s_0 = A\cos(2\pi F_0 t + \pi)$ to represent a "0" or $s_1 = A\cos(2\pi F_0 t)$ to represent a "1", where $A > 0$. The receiver that is used to decode the transmission is shown in the figure below. The receiver that is used to decode the transmission is shown in the figure below.



The input to the receiver is the noise corrupted signal or $x(t) = s_i(t) + w(t)$, where $w(t)$ represents the channel noise. It can be shown that in the absence of noise, the output of the lowpass filter is

$$\zeta = \begin{cases} -\frac{A}{2} & \text{for } a\ ''0'' \\ \frac{A}{2} & \text{for } a\ ''1'' \end{cases}$$

The receiver decides a "1" was transmitted if $\zeta > 0$ and a "0" if $\zeta \leq 0$. To model the channel noise we assume that the actual value of ζ observed is

$$\zeta = \begin{cases} -\frac{A}{2} + W & \text{for } a\ ''0'' \\ \frac{A}{2} + W & \text{for } a\ ''1'' \end{cases}$$

where W is the Gaussian random variable, $W \sim \mathrm{N}(0,1)$.

1. Analytically determine the probability of error P_e, and plot it versus A.
2. Use computer simulation to plot P_e versus A on the same graph as in 1 above.
3. How should one choose A so that the error probability does not exceed 0.01 (use the analytical result)?

1.1 Analytical Solution

The first goal is to analytically determine the probability of error P_e, and plot it versus A. First, we will declare the symbolic variables of the system, where A is the signal amplitude, W is the channel noise, and x is a generic variable. Then, we initialize the probabilities of a 0 or 1 is being sent.

```
syms A W x
P_0 = 0.5; % Probability that a "0" is sent
P_1 = 0.5; % Probability that a "1" is sent
```

Next, we declare the function of the Normal Gaussian Distribution which has the form of,

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{\left(-\frac{x^2}{2}\right)}$$

and substitute our signal functions (with noise) for the variable x.

```
Guass_Norm_pdf = 1/sqrt(2*pi) * exp((-(x)^2)/2); % Normal Gaussian Distribution func.

Zeta_0_pdf = subs(Guass_Norm_pdf, x, (A/2 + W)); % PDF of a "0" being read
Zeta_1_pdf = subs(Guass_Norm_pdf, x, (A/2 - W)); % PDF of a "1" being read
```

This yeilds the following two results for the probability distributions of a "0" or "1" being sent,

$$f_0 = \frac{1}{\sqrt{2\pi}} e^{\left(-\frac{\left(-\frac{A}{2}+w\right)^2}{2}\right)}, \quad \text{for a "0"}$$

$$f_1 = \frac{1}{\sqrt{2\pi}} e^{\left(-\frac{\left(\frac{A}{2}+w\right)^2}{2} \right)}, \quad \text{for a "1"}$$

Subsequently, the cdf, or probability of error given that the input is a "0" or "1" can be calculated by integration

$$P(\text{error}|0) = \int_{-\infty}^{\infty} f_0(w)\text{dw}$$

$$P(\text{error}|1) = \int_{-\infty}^{\infty} f_1(w)\text{dw}$$

```
P_e_given_0 = 1 - int(Zeta_0_pdf,W,-inf,0); % CDF of an error given a "0"
P_e_given_1 = int(Zeta_1_pdf,W,-inf,0);     % CDF of an error given a "1"
```

Finally, the analytical solution to the total probability of error as a function of A can be expressed as

$P_e = P(\text{error}|0)(P(0)) + P(\text{error}|1)(P(1))$

```
P_e_analytical = vpa((P_e_given_0*P_0 + P_e_given_1*P_1), 4) % Probability of error as
```

P_e_analytical = $0.5 - 0.5\,\text{erf}\,(0.3536\,A)$

```
                                                              % function of A
```

However, this result can be expressed in a more neat manner as follows,

$$P_e = \frac{1}{2}\left(1 - \text{erf}\left(\frac{\sqrt{2}}{4}A\right)\right)$$

1.2 Computer Simulation

The second goal is use a computer simulation to plot P_e versus A on the same graph as in 1 above. First, we declare the range of values of A that we would like to use.

```
Astep = 20;                          % Step size of A values
Aspan = [-5, 5];                   % The minimum and maximum values for A
A_values = [Aspan(1):range(Aspan)/Astep:Aspan(2)];  % The array of A values
errors = zeros(length(A_values), 1);   % An empty matrix containing the
                                   % simulated number of errors for given A value
```

Then, we calculate the percent of error for each value of A, for n number of samples. First, we calculate the probability of error P_e as a function of signal amplitude A for sample sizes of 100.

```
n = 100; % Sample Size
% Iterate through each value of A
for k = 1:length(A_values)
    dist = normrnd(A_values(k)/2, 1, [1 n]); % Distribution of read values for A
    % Iterate through each sample of read values.
```

```
    for m = 1:n
        % Determine if the received value is an error
        if dist(m) <= 0
            errors(k) = errors(k) + 1; % Count the number of errors.\
        end
    end
end

P_e_simulation_100 = errors/n; % Probability of error vs A for n=100
```

Next, we calaculate the probability of error P_e as a function of signal amplitude A for sample sizes of 100,000.

```
n = 1000000; % Sample Size

% Iterate through each value of A
for k = 1:length(A_values)
    dist = normrnd(A_values(k)/2, 1, [1 n]); % Distribution of read values for A
    % Iterate through each sample of read values.
    for m = 1:n
        % Determine if the received value is an error
        if dist(m) <= 0
            errors(k) = errors(k) + 1; % Count the number of errors.\
        end
    end
end

P_e_simulation_100k = errors/n; % Probability of error vs A for n=100000.
```

Finally, we can plot the results.

```
fig1 = figure; % Create a new figure
hold on
% Plot the analytical solution
fplot(P_e_analytical, 'Color', [0 0 0])

% Plot the simulation for n=100
plot(A_values, P_e_simulation_100, '.', 'MarkerSize', 12 ...
                        , 'Color', [0.8750 0.2780 0.3840])

% Plot the simulation for n=100000
plot(A_values, P_e_simulation_100k, '^', 'MarkerSize', 5 ...
                         , 'Color', [0 0.4470 0.7410] ...
                         , 'MarkerFaceColor', [0 0.4470 0.7410])
hold off

% Label the graph appropriately
legend('Analytical', 'Simulation, n=100', 'Simulation, n=100k')
title("P_e versus A")
xlabel("Signal Magnitude, {\itA} [-]")
ylabel("Probability of Error, {\itP_e} [-]")
```

```
shg % Show the Figure
```

From the plot above, we can see the the numerical and analytical solutions agree with one another.

From the trend, we can see that the probability of error, P_e, *decreases* for large values of A. This is expected, as if we increase the magnitude of the input the signal, A, the effect of the that the random Gaussian noise, W, has on the signal decreases. As the limit of the magnitude of the input signal approaches infinity, the probability of a error goes to zero.

$$\lim_{A \to \infty} P_e(A) = 0$$

Alternatively, when the magnitude of the input signal, A, is inverted (negative), the probability of error, P_e, increases. This can be thought of as *purposefully* sending the wrong bit to the receiver.

$$\lim_{A \to -\infty} P_e(A) = 1$$

Perhaps the most interesting feature of this plot is the probability of error when A is equal to zero (no purposeful signal). We can see that the expected value of error approaches 0.5. At this point, the probability of error has the same distribution as the random variable, W, which is a Normal Guassian distribution (centered at 0 with a standard deviation of 1).

1.3 Discussion

Q: How should one choose A so that the error probability does not exceed 0.01 (use the analytical result)?

In order to choose a value of A such that the error probability does not exceed 0.01, the normal cdf can be used. This desired result can be expressed by the following equation,

$$P_e(A) \leq 0.01$$

where the probability of error for a value of A can be expressed by the solution in part 1.1,

$$\frac{1}{2}\left(1 - \operatorname{erf}\left(\frac{\sqrt{2}}{4}A\right)\right) \leq 0.01$$

then, multiplying both sides by 2

$$1 - \operatorname{erf}\left(\frac{\sqrt{2}}{4}A\right) \leq 0.02$$

then, subtracting 1 from both sides while keeping A on the left hand side,

$$\operatorname{erf}\left(\frac{\sqrt{2}}{4}A\right) \leq 0.98$$

and taking the inverse error function of both sides,

$$\frac{\sqrt{2}}{4}A \leq \operatorname{erf}^{-1}(0.98)$$

isolating A to the left hand side of the equation

$$A \leq 2\sqrt{2}\,\mathrm{erf}^{-1}(0.98)$$

and finally, evaluting the inverse error function and square root for the numerical approximation,

$$A \geq 4.653$$

Likewise, this analytical solution can be solved simply by the following line of code,

```
A_min = vpa(solve(0.01 == P_e_analytical,A), 4) % Minimum value of A such P_e <= 0.01
```

```
A_min = 4.653
```

Graphically, we can show this point by drawing a line at P_e is equal to 0.01, and finding the corresponding signal magnitude, A, on the plot.

```
fig2 = fig1;
hold on
xlim([2 5]);
ylim([0 0.05])
legend('Analytical', 'Simulation, n=100', 'Simulation, n=100k', "Location","northwest")
```

Part 2. COVID-19 Data Analysis

Problem Statement

In this assignment, you will analyze the data to understand the Coronavirus pandemic, in particular, you will study the factors that might impact the mortality rate of COVID-19. You can find the dataset provided by *Our World in Data* here. It provides daily numbers on confirmed cases and deaths, testings, vaccinations, and other metrics for over 100 countries. For this study, consider the following data: *new_deaths_per_million* for the month of September 2021, *new_tests_per_thousand* for the same period, *people_vaccinated_per_hundred*, *aged_70_older*, *gdp_per_capita, cardiovasc_death_rate, diabetes_prevalence*. If a country misses any of the data, remove that country in this study.

1. Plot the distribution of death rates for all countries. Compute the average death rates and the variance (Don't use MATLAB built-in functions). What do these numbers tell us? Describe anything else you observe from the data.
2. Study all other metrics. Plot their distributions, compute and interpret the average values and variances.
3. Compute the correlation coefficients between the death rates and other metrics. What factor(s) affect the mortality rate the most? Plot the scatter diagram to support your argument. Describe anything else you observe from the data.

Import and Filtering Data

First, we must import the most recent data from the file of comma separated variables.

```
raw_covid_data = readtable('owid-covid-data.csv'); % Table of raw covid data
```

Next, we can identify a list of unique country names

```
unique_countries = unique(raw_covid_data.location); % Unique country names
```

Now comes the tricky part. For this study, we reject all countries that miss any of the data. That is, each country must have a at least one distinct value for every random variable under consideration. To do this, we create an empty array which will contain the values of all unique countries that fit this rule. Next, check whether the country has at least one data point for every random variable. If the country does not contain an empty or null array of values for each random variable, we accept it and append it to a list of countries which fit this category.

```
countries = cell(1);
% Aquire data for each country with valid data
for k = 1:length(unique_countries)
    % Filter by month and valid values
    if or(isempty(raw_covid_data.new_deaths_per_million( ...
            ...
            string(raw_covid_data.location) == unique_countries(k) ...
                ...
                & raw_covid_data.date >= '1-Sep-2021' ...
                & raw_covid_data.date <= '30-Sep-2021')) ...
       , all(isnan(raw_covid_data.new_deaths_per_million( ...
            string(raw_covid_data.location) == unique_countries(k) ...
                & raw_covid_data.date >= '1-Sep-2021' ...
                & raw_covid_data.date <= '30-Sep-2021'))))
        continue
    % Filter by month and valid values
    elseif or(isempty(raw_covid_data.new_tests_per_thousand(string(raw_covid_data.location) ==
                                                    & raw_covid_data.date >= '1-Sep-2021' ...
                                                    & raw_covid_data.date <= '30-Sep-2021')) .
           , all(isnan(raw_covid_data.new_tests_per_thousand(string(raw_covid_data.location) ==
                                                    & raw_covid_data.date >= '1-Sep-2021' ...
                                                    & raw_covid_data.date <= '30-Sep-2021'))))
        continue
    % Filter by valid values
    elseif or(isempty(raw_covid_data.people_vaccinated_per_hundred( ...
                string(raw_covid_data.location) == unique_countries(k))) ...
           , isnan(raw_covid_data.people_vaccinated_per_hundred( ...
                string(raw_covid_data.location) == unique_countries(k))))
       continue
    % Filter by valid values
    elseif or(isempty(raw_covid_data.aged_70_older( ...
                string(raw_covid_data.location) == unique_countries(k))) ...
           , isnan(raw_covid_data.aged_70_older( ...
                string(raw_covid_data.location) == unique_countries(k))))
        continue
    % Filter by valid values
    elseif or(isempty(raw_covid_data.gdp_per_capita( ...
                string(raw_covid_data.location) == unique_countries(k))) ...
           , isnan(raw_covid_data.gdp_per_capita( ...
                string(raw_covid_data.location) == unique_countries(k))))
```

```
        continue
    % Filter by valid values
    elseif or(isempty(raw_covid_data.cardiovasc_death_rate( ...
                string(raw_covid_data.location) == unique_countries(k))) ...
           , isnan(raw_covid_data.cardiovasc_death_rate( ...
                string(raw_covid_data.location) == unique_countries(k))))
        continue
    % Filter by valid values
    elseif or(isempty(raw_covid_data.diabetes_prevalence( ...
                string(raw_covid_data.location) == unique_countries(k))) ...
           , isnan(raw_covid_data.diabetes_prevalence( ...
                string(raw_covid_data.location) == unique_countries(k))))
        continue
    end
    % Append country name to list of countries with valid data
    countries(end+1,1) = unique_countries(k);
end
countries = string(countries(2:end)); % Remove the first, null index
```

Now, we have a list of countries which contain data for each random variable under consideration.

2.1 Distribution of Death Rates

Before we plot the distribution of death rates for all countries, we first determine the distribution of death rates for each country. For each country, we take the *average* death rates per million people. This gives us a distribution of average death rates per million people per country. Subsequently, the average value value of death rates per country is equal to the total average death rates.

Since we will be aquiring the distribution of other random variables, we can condense the collection of these distribution into a single **for** loop. First, we initialize empty arrays for each variable which are the lengths of the number of country. Thus, the total collection of variables per country makes up our distribution of random variables.

```
% Distributions of random variables
dist_avg_deaths_per_million = zeros(length(countries), 1);
dist_avg_tests_per_thousand = zeros(length(countries), 1);
dist_people_vaccinated_per_hundred = zeros(length(countries), 1);
dist_avg_aged_70_older = zeros(length(countries), 1);
dist_avg_gdp_per_capita = zeros(length(countries), 1);
dist_avg_cardiovasc_death_rate = zeros(length(countries), 1);
dist_avg_diabetes_prevalence = zeros(length(countries), 1);
```

One important point to note is that we will be computing the average (expected) value of *each* random variable for each country.

```
for k = 1:length(countries)
    new_deaths_per_million = raw_covid_data.new_deaths_per_million( ...
                                string(raw_covid_data.location) == countries(k) ...
                                & raw_covid_data.date >= '1-Sep-2021' ...
                                & raw_covid_data.date <= '30-Sep-2021' ...
                                & ~isnan(raw_covid_data.new_deaths_per_million));
```

```
    new_tests_per_thousand = raw_covid_data.new_tests_per_thousand( ...
                                string(raw_covid_data.location) == countries(k) ...
                                & raw_covid_data.date >= '1-Sep-2021' ...
                                & raw_covid_data.date <= '30-Sep-2021' ...
                                & ~isnan(raw_covid_data.new_tests_per_thousand));

    dist_people_vaccinated_per_hundred(k) = max(raw_covid_data.people_vaccinated_per_hundred( .
                                           string(raw_covid_data.location) == countries(k) ...
                                           & ~isnan(raw_covid_data.people_vaccinated_per_hundred)

    aged_70_older = raw_covid_data.aged_70_older( ...
                    string(raw_covid_data.location) == countries(k) ...
                    & ~isnan(raw_covid_data.aged_70_older));

    gdp_per_capita = raw_covid_data.gdp_per_capita( ...
                     string(raw_covid_data.location) == countries(k) ...
                     & ~isnan(raw_covid_data.gdp_per_capita));

    cardiovasc_death_rate = raw_covid_data.cardiovasc_death_rate( ...
                            string(raw_covid_data.location) == countries(k) ...
                            & ~isnan(raw_covid_data.cardiovasc_death_rate));

    diabetes_prevalence = raw_covid_data.diabetes_prevalence( ...
                          string(raw_covid_data.location) == countries(k) ...
                          & ~isnan(raw_covid_data.diabetes_prevalence));


    dist_avg_deaths_per_million(k) = first_moment(new_deaths_per_million);
    dist_avg_tests_per_thousand(k) = first_moment(new_tests_per_thousand);
    dist_avg_aged_70_older(k) = first_moment(aged_70_older);
    dist_avg_gdp_per_capita(k) = first_moment(gdp_per_capita);
    dist_avg_cardiovasc_death_rate(k) = first_moment(cardiovasc_death_rate);
    dist_avg_diabetes_prevalence(k) = first_moment(diabetes_prevalence);
end

distributions = [dist_avg_deaths_per_million, dist_avg_tests_per_thousand, dist_people_vaccinat
                 dist_avg_aged_70_older, dist_avg_gdp_per_capita, dist_avg_cardiovasc_death_rat
random_variable_names = {'Avg. New Deaths per Million', 'Avg. New Tests per Thousand', 'People
                         'Avg. Aged 70 or Older', 'Avg. GDP per Capita', 'Avg. Cardiocascular D
```

```
random_variable_names = 1×7 cell
'Avg. New Deaths per Million'  'Avg. New Tests per Thousand'  'People Vaccinated ···
```

```
table_distributions = array2table(distributions, 'RowNames',countries, ...
                                  'VariableNames', random_variable_names)
```

```
table_distributions = 97×7 table
                                                                                    ...
```

	Avg. New Deaths per Million	Avg. New Tests per Thousand
1 Argentina	2.4609	0.6032
2 Armenia	5.3345	1.9166
3 Australia	0.3864	8.3364

	Avg. New Deaths per Million	Avg. New Tests per Thousand
4 Austria	0.8737	39.0196
5 Azerbaijan	2.8986	1.5741
6 Bahamas	14.8647	1.0742
7 Bahrain	0.0191	10.6501
8 Bangladesh	0.2635	0.1647
9 Belarus	1.2816	2.2106
10 Belgium	0.6363	3.8349
11 Bhutan	0	2.5440
12 Bolivia	0.7974	0.4376
13 Bosnia and Herzegovina	8.2020	1.0066
14 Canada	0.8098	2.4236
15 Cape Verde	1.5425	1.6684
16 Chile	0.9212	2.6300
17 Colombia	0.8804	0.8909
18 Costa Rica	5.7079	1.3645
19 Cote d'Ivoire	0.2257	0.1502
20 Croatia	2.4990	2.1905
21 Cyprus	1.5764	61.0986
22 Czechia	0.1708	6.7030
23 Denmark	0.4128	7.0968
24 Ecuador	0.9653	0.1313
25 Equatorial Guinea	0.4828	1.5000
26 Estonia	1.6350	4.2354
27 Ethiopia	0.2564	0.0662
28 Finland	0.3004	2.8847
29 France	1.2571	8.0229
30 Gabon	0.2926	12.0665
31 Georgia	12.8818	12.9405
32 Greece	3.8763	15.1402
33 Guatemala	2.9918	0.5879
34 Hungary	0.4567	1.6902
35 Iceland	0	5.0044
36 India	0.2230	1.1294
37 Indonesia	1.0754	0.5718

	Avg. New Deaths per Million	Avg. New Tests per Thousand
38 Iran	4.9529	1.3034
39 Iraq	1.1575	0.6794
40 Ireland	1.0503	4.6724
41 Israel	2.7229	14.4826
42 Italy	0.9387	4.6547
43 Jamaica	3.9348	0.6014
44 Japan	0.4188	0.7644
45 Jordan	0.9965	2.8439
46 Kuwait	0.2310	3.5491
47 Laos	0.0181	0.7075
48 Libya	1.9353	0.8165
49 Lithuania	5.3535	6.8588
50 Luxembourg	0.2625	5.3090
51 Madagascar	0.0023	0.0240
52 Maldives	0.3066	8.8657
53 Malta	1.0364	6.1961
54 Mexico	4.6525	0.1770
55 Mongolia	3.4041	2.8868
56 Morocco	1.4441	0.5961
57 Mozambique	0.0548	0.0570
58 Myanmar	1.4269	0.4219
59 Namibia	1.7519	0.5703
60 Nepal	0.4324	0.3304
61 New Zealand	0.0069	2.6148
62 Norway	0.2867	3.8635
63 Pakistan	0.2808	0.2377
64 Palestine	2.6679	1.9324
65 Panama	1.2705	1.6157
66 Paraguay	1.9900	0.3046
67 Peru	1.1031	0.3118
68 Philippines	1.4547	0.6633
69 Poland	0.2689	0.9813
70 Portugal	0.7605	5.0948
71 Qatar	0.0455	1.8231

	Avg. New Deaths per Million	Avg. New Tests per Thousand
72 Romania	4.3061	2.8210
73 Rwanda	0.4620	0.9697
74 Saudi Arabia	0.1924	1.3632
75 Senegal	0.1802	0.1158
76 Slovakia	0.5432	5.0784
77 Slovenia	1.7798	2.1250
78 South Africa	2.9786	0.6864
79 South Korea	0.1330	0.7526
80 Sri Lanka	5.7697	0.4742
81 Sweden	0.5643	2.9517
82 Switzerland	0.7190	4.0413
83 Thailand	2.4484	0.7179
84 Timor	1.2401	0.9910
85 Togo	0.1731	0.2010
86 Trinidad and Tobago	4.5367	1.0402
87 Tunisia	4.0187	1.3501
88 Turkey	2.7896	3.8696
89 Uganda	0.1019	0.0752
90 Ukraine	2.2821	0.6410
91 United Arab Emirates	0.1867	30.6908
92 United Kingdom	2.0447	14.8517
93 United States	5.6741	4.5165
94 Uruguay	0.2200	2.8201
95 Vietnam	2.7968	3.6790
96 Zambia	0.0813	0.3169
97 Zimbabwe	0.4506	0.2791

Part 1. Death Rates in September

```
fig3 = figure
```

```
fig3 =
  Figure (98) with properties:

      Number: 98
        Name: ''
       Color: [0.9400 0.9400 0.9400]
    Position: [680 558 560 420]
       Units: 'pixels'
```

Show all properties

```
histogram(dist_avg_deaths_per_million, round(sqrt(length(countries))))
title('Average new death rates per million by country for the month of September')
ylabel('Number of countries')
xlabel('Avg. New deaths per million')
```

Part 2. Analysis of Other Metrics

```
fig4 = figure
```

```
fig4 =
  Figure (100) with properties:

      Number: 100
        Name: ''
       Color: [0.9400 0.9400 0.9400]
    Position: [680 558 560 420]
       Units: 'pixels'

  Show all properties
```

```
set(gcf, 'position', [0 0 800 500])
subplot(2, 3, 1)
histogram(table_distributions.("Avg. New Tests per Thousand"), round(sqrt(length(countries))))
xlabel('Avg. New Tests per Thousand')
subplot(2, 3, 2)
histogram(table_distributions.("People Vaccinated Per Hundred"), round(sqrt(length(countries))))
xlabel('Vaccinated Per Hundred')
subplot(2, 3, 3)
histogram(table_distributions.("Avg. Aged 70 or Older"), round(sqrt(length(countries))))
xlabel('Avg. Aged 70 or Older')
subplot(2, 3, 4)
histogram(table_distributions.("Avg. GDP per Capita"), round(sqrt(length(countries))))
xlabel('Avg. GDP per Capita')
subplot(2, 3, 5)
histogram(table_distributions.("Avg. Cardiocascular Death Rate"), round(sqrt(length(countries))))
xlabel('Avg. Cardiocascular Death Rate')
subplot(2, 3, 6)
histogram(table_distributions.("Avg. Diabetes Prevalence"), round(sqrt(length(countries))))
xlabel('Avg. Diabetes Prevalence')
```

```
table_averages = array2table(first_moment(table_distributions), 'RowNames', {'Global Average'},
                                                                'VariableNames', random_variabl
```

```
table_averages = 1×7 table
                                                                        ...
```

	Avg. New Deaths per Million	Avg. New Tests per Thousand
1 Global Average	1.8016	4.1216

```
table_variences = array2table(second_moment(table_distributions), 'RowNames', {'Global Varience
                                                                'VariableNames', random_variable_na
```

```
table_variences = 1×7 table
```

...

	Avg. New Deaths per Million	Avg. New Tests per Thousand
1 Global Varience	5.9724	67.0796