Probability Distribution Problems

DR. MD. ZAHIDUL ISLAM
PROFESSOR
DEPT. OF INFORMATION AND
COMMUNICATION TECHNOLOGY
ISLAMIC UNIVERSITY, KUSHTIA-7003
BANGLADESH

Problem 1: Simulating a Normal Distribution

Objective: Generate and plot a normal distribution with specified mean and standard deviation.

```
% Parameters

mu = 0; % Mean

sigma = 1; % Standard Deviation


% Generate a normal distribution

x = -3*sigma:0.1:3*sigma;

y = normpdf(x, mu, sigma);


% Plot

figure;

plot(x, y);

title('Normal Distribution');

xlabel('Value');

ylabel('Probability Density');
```

Problem 2: Calculating Binomial Probabilities

Objective: Calculate and plot the probabilities for a binomial distribution.

```
% Parameters

n = 10; % Number of trials

p = 0.5; % Probability of success
```

```
% Calculate binomial probabilities
x = 0:n;
y = binopdf(x, n, p);

% Plot
figure;
bar(x, y);
title('Binomial Distribution');
xlabel('Number of Successes');
ylabel('Probability');
```

Problem 3: Poisson Distribution

Objective: Plot the Poisson distribution for different mean values.

```
% Parameters
lambda = [2, 4, 6]; % Different mean values

% Generate and plot Poisson distributions
figure;
for i = 1:length(lambda)
  x = 0:15;
  y = poisspdf(x, lambda(i));
  subplot(1, length(lambda), i);
  bar(x, y);
```

```
    title(['Poisson Distribution with \lambda = ', num2str(lambda(i))]);
    xlabel('Number of Events');
    ylabel('Probability');
end
```

Problem 4: Exponential Distribution

Objective: Generate an exponential distribution and plot its probability density function.

```
% Parameter
lambda = 1; % Rate parameter

% Generate exponential distribution
x = 0:0.1:10;
y = exppdf(x, 1/lambda);

% Plot
figure;
plot(x, y);
title('Exponential Distribution');
xlabel('Value');
ylabel('Probability Density');
```

Problem 5: Uniform Distribution

Objective: Simulate and visualize a uniform distribution.

```
% Parameters
a = 0; % Lower bound
b = 1; % Upper bound

% Generate uniform distribution
x = a:0.01:b;
y = unifpdf(x, a, b);

% Plot
figure;
plot(x, y);
title('Uniform Distribution');
xlabel('Value');
ylabel('Probability Density');
```

Problem 6: Cumulative Distribution Function (CDF)

Objective: Plot the CDF of a normal distribution.

```
% Parameters for normal distribution
mu = 0; % Mean
sigma = 1; % Standard deviation

% Generate CDF
x = -3*sigma:0.1:3*sigma;
y = normcdf(x, mu, sigma);
```

```
% Plot
figure;
plot(x, y);
title('CDF of Normal Distribution');
xlabel('Value');
ylabel('Cumulative Probability');
```

Problem 7: Working with Random Variables

Objective: Generate random numbers from a normal distribution and calculate their statistics.

```
% Generate random numbers
num_samples = 1000;
samples = normrnd(0, 1, [num_samples, 1]);

% Calculate statistics
mean_val = mean(samples);
std_dev = std(samples);

% Display results
fprintf('Mean: %.2f\n', mean_val);
fprintf('Standard Deviation: %.2f\n', std_dev);
```

Basic Statistics with MATLAB

1. Calculating Mean, Median, and Mode

Objective: Find the mean, median, and mode of a given data set.

```
% Data
data = [15, 9, 26, 13, 14, 12, 22, 19];

% Mean
mean_val = mean(data);

% Median
median_val = median(data);

% Mode
mode_val = mode(data);

% Display results
fprintf('Mean: %.2f\n', mean_val);
fprintf('Median: %.2f\n', median_val);
fprintf('Mode: %.2f\n', mode_val);
```

2. Standard Deviation and Variance

Objective: Compute the standard deviation and variance of a data set.

```
% Data
data = [15, 9, 26, 13, 14, 12, 22, 19];
```

```
% Standard Deviation
std_dev = std(data);

% Variance
variance = var(data);

% Display results
fprintf('Standard Deviation: %.2f\n', std_dev);
fprintf('Variance: %.2f\n', variance);
```

3. Linear Correlation Coefficient

Objective: Determine the linear correlation coefficient (Pearson's r) between two sets of data.

```
% Data
data_x = [1, 2, 3, 4, 5];
data_y = [2, 4, 5, 4, 5];

% Correlation Coefficient
corr_coeff = corrcoef(data_x, data_y);

% Display result
fprintf('Correlation Coefficient: %.2f\n', corr_coeff(1,2));
```

4. Histogram Plotting

Objective: Create a histogram to visualize the distribution of a data set.

```
% Data
data = [15, 9, 26, 13, 14, 12, 22, 19];

% Plot Histogram
figure;
histogram(data);
title('Data Distribution');
xlabel('Value');
ylabel('Frequency');
```

5. Boxplot for Data Distribution

Objective: Generate a boxplot to observe the spread and skewness of data.

```
% Data
data = [15, 9, 26, 13, 14, 12, 22, 19];

% Boxplot
figure;
boxplot(data);
title('Boxplot of Data');
```

6. Scatter Plot for Two Variables

Objective: Create a scatter plot to visualize the relationship between two variables.

```
% Data
data_x = [1, 2, 3, 4, 5];
data_y = [2, 4, 5, 4, 5];
```

```
% Scatter Plot
figure;
scatter(data_x, data_y);
title('Scatter Plot');
xlabel('X-axis');
ylabel('Y-axis');
```

7. Generating Random Data and Analyzing

Objective: Generate random data following a normal distribution and analyze it.

```
% Generate random data
data = normrnd(0, 1, [100, 1]); % 100 random numbers from N(0,1)
% Mean and Standard Deviation
mean_val = mean(data);
std_dev = std(data);

% Display results
fprintf('Mean of Random Data: %.2f\n', mean_val);
fprintf('Standard Deviation of Random Data: %.2f\n', std_dev);

% Plot Histogram
figure;
histogram(data);
title('Histogram of Random Data');
```

Regression Analysis

Regression analysis is a powerful statistical method for examining the relationship between a dependent variable and one or more independent variables. Here's how you can perform linear regression analysis in MATLAB:

Simple Linear Regression

Simple linear regression involves a single independent variable. Let's say we have some data x and y, and we want to fit a line y = a*x + b.

```
% Sample Data

x = [1, 2, 3, 4, 5];

y = [2, 3, 4, 6, 5];


% Perform linear regression

p = polyfit(x, y, 1); % p(1) is slope, p(2) is intercept


% Create a linear model

y_fit = polyval(p, x);


% Plot

figure;

plot(x, y, 'o'); % original data

hold on;

plot(x, y_fit, '-'); % fitted line
```

```
title('Simple Linear Regression');
xlabel('x');
ylabel('y');
legend('Data', 'Fitted line');
```

Multiple Linear Regression

In multiple linear regression, we predict a dependent variable based on multiple independent variables.

```
% Sample Data
X = [1 2 3; 2 3 4; 3 4 5; 4 5 6; 5 6 7]; % Each row is an observation
y = [2; 3; 4; 6; 5]; % Dependent variable
% Add a column of ones to X for the intercept
X = [ones(size(X, 1), 1) X];
% Perform regression
b = regress(y, X); % Returns the regression coefficients

% Predicted values
y_pred = X * b;

% Display the coefficients
disp('Coefficients (including intercept):');
disp(b);

% Plot - only practical if you have 1 or 2 independent variables
% For more variables, consider 3D plots or partial regression plots
```

Polynomial Regression

Polynomial regression fits a nonlinear relationship between the value of x and the corresponding conditional mean of y.

```
% Sample Data

x = [1, 2, 3, 4, 5];

y = [2, 4, 6, 8, 10];

% Polynomial degree

degree = 2;

% Perform polynomial regression

p = polyfit(x, y, degree);


% Create a polynomial model

x_fit = linspace(min(x), max(x), 100); % 100 points for a smoother plot

y_fit = polyval(p, x_fit);


% Plot

figure;

plot(x, y, 'o'); % original data

hold on;

plot(x_fit, y_fit, '-'); % fitted polynomial

title('Polynomial Regression');

xlabel('x');

ylabel('y');

legend('Data', 'Fitted polynomial');
```