|  |  |  |
| --- | --- | --- |
| Exam 2 will cover: | | |
| LOs 01.00 – 09.00 (Exam 1) | 10.00 Cumulative Distribution Plots  11.00 User-Defined Functions | 12.00 Regression  13.00 Function Discovery |

## 10.00 Create and interpret cumulative distribution plots

| Learning Objective | Evidence |
| --- | --- |
| 10.01 Compute the relative fractional values given the bin intervals | Call the histogramRight function (with the correct input arguments) to generate the histogram properties  Determine the frequencies for each bin in a right-bin inclusive histogram  Determine the total number of data points accounted for in the overall histogram  Calculate the fractional values by dividing the frequency in each bin by the total number of data points accounted for in the overall histogram |
| 10.02 Compute cumulative fractional values given the bin intervals | Correct syntax for cumsum function  Perform the cumulative sum to get vector of cumulative fractional values  Start the cumulative sum vector at 0 |
| 10.03 Create a cumulative distribution plot using the companion histogram’s bin right edges | Correct syntax for the plot command: plot(x, y, ‘line/marker formatting’)  Independent variable (x) is the bin edge values from a right-bin inclusive histogram  Dependent variable (y) is the cumulative fractional values corresponding to the right-bin inclusive histogram  Correct use of data markers and lines: data markers (for the bin edges) with an overlaid line (for the model) |
| 10.04 Format a cumulative distribution plot for technical presentation | Correct syntax for title  Correct syntax for xlabel  Correct syntax for ylabel  Descriptive title that references the problem context and x-variable data  Clear x-axis label with units  Clear y-axis label that is cumulative fractional value  y-axis scale range of 0 to 1  x-axis scales that match each other, when using subplots to compare data  Color and marker/line style(s) that are as specified or distinctive (when multiple data sets)  Proper formatting of a legend, when multiple data sets and/or models  Gridlines |
| 10.05 Determine the likelihood of event occurrences using a cumulative distribution plot | Determine the likelihood of an occurrence of a value:  less than specified criteria  greater than specified criteria  between specified criteria reading the fractional value for given data point  Clear explanation of how the likelihood is determined |
| 10.06 Estimate and/or describe the process for determining the characteristics of the underlying data set from a cumulative distribution plot | Estimate the median of the data by reading the CDP at 0.5 cumulative fractional value (within 2% of solution answer)  Estimate the range of the data by reading the CDP at 0 and 1 cumulative fractional values (within 2% of solution answer)  Clear description of a process for determining the median  Clear description of a process for determining the range |
| 10.07 Determine the data distribution type from the shape of a cumulative distribution plot | Identify the shape of the distribution (uniform, unimodal, bimodal, normal, etc)  Justify shape identification  Identify the skew of the distribution (positive, negative, undefined, etc)  Justify skew identification |
| 10.08 Draw inferences from the analysis of data with evidence from a cumulative distribution | For a given data set and a problem context, appropriately use a cumulative distribution as described in 10.05 – 10.07 to draw conclusions. |

## 11.00 Create and execute a user-defined function

| Learning Objective | Evidence |
| --- | --- |
| 11.01 Describe at least two reasons why MATLAB user-defined functions as opposed to scripts are used | Recognition that UDFs enable one to create an easily re-usable piece of code  Recognition that UDFs can be shared with others without them having to know what variables were used by the author  Recognition that UDFs enable a larger program to broken into smaller parts that can be more easily tested & debugged  Recognition that UDFs allow team members to work on separate parts of a larger program with less coordination |
| 11.02 Describe three ways a user-defined function is different from a script | Recognition that the first line of a UDF is the function definition line; the first line of a script can be any executable line of code  Recognition that the variables created in a UDF are not available in the Workspace; all variables created in script are available in the Workspace  Recognition that a UDF must be called from the command line or from within another function or script; the green run button will not work for a UDF that has input arguments |
| 11.03 Create a user-defined function that adheres to programming standards | Help lines contain input and output argument definitions, with units as appropriate  Help lines contain concise description of the program  Help lines show the call to the function  Complete programmer and contributor information in the header (names and emails)  Complete problem details including assignment number, problem number  Code items are in the correct section (e.g. Initialization, Calculations, …)  Computed values are assigned to variables  Code blocks have explanatory comments  Variables have commented definitions and units  Minimal use of hardcoding |
| 11.04 Construct an appropriate function definition line | Correct syntax for a function:   * function [output1,…,outputN] = function\_name(input1,…,inputM) * Function starts with the keyword function * Order is output arguments, equal sign, function name, input arguments * Functions with no inputs have no input list; use of ( ) is optional * Functions with no output arguments have no output list and no equal sign * Multiple output arguments are listed inside square brackets, separated by spaces or commas * Multiple input arguments are listed inside parentheses and are comma-separated   Function definition line is the first line in the function file (above help lines)  Function file name matches the function name in the definition line  Input arguments must meet the problem specifications (with no extraneous input arguments) or be appropriate for the purpose of the function  Output arguments must meet the problem specifications or be appropriate for the purpose of the function  Output arguments must be assigned within the function code |
| 11.05 Match the variables names used in the function definition line to those used in the function code | All input arguments are used in the code  All input arguments necessary to perform computations are provided in the function definition  Input arguments are not overwritten (e.g. by hardcoded values) before being used in calculations  All output arguments are appropriately assigned in the function code |
| 11.06 Execute a user-defined function | Correct syntax to call a function:   * [output1,…,outputN] = function\_name(input1,…,inputM) * Call does not contain keyword function * Order is output arguments, equal sign, function name, input arguments, with output arguments and equal sign being optional for a no-output function * Functions with no inputs have no input list; use of ( ) is optional * Functions with no output arguments have no output list and no equal sign * Multiple output arguments are listed inside square brackets, separated by spaces or commas * Multiple input arguments are listed inside parentheses and are comma-separated   Calls the correct function filename  Number of input arguments matches the number required by the function  Input argument list corresponds to the function’s expected inputs  Number of output argument(s) matches the number required by the function  Output argument list corresponds to the function’s expected outputs |
| 11.07 Create test cases to evaluate a user-defined function | Running the UDF with a variety of reasonable values for each input argument to ensure no computation or execution errors occur  Running the UDF with both scalar and array input arguments to ensure no errors occur |
| 11.08 Convert a script to a user-defined function | First line of code is a function definition line  Replacement of script header with function header  Removal of hardcoded variable assignments for all variables in the input argument list |
| 11.09 Track the passing of information to and from a user-defined function | Being able to take given input argument values to one UDF and manually track the value of all computed and passed output arguments and input arguments through a series of UDFs linked through function calls |
| 11.10 Break a problem into a series of sub-functions | Being able to take a complex task and break it into a series of unique UDFs that are each purposeful and easy to test and debug |
| 11.11 Coordinate the passing of information between functions | Call to a user-defined function occurs in the proper function or script  Variables passed into a user-defined function are defined prior to calling the user-defined function  User-defined functions are called in the order necessary to complete the coding task  No use of global variables (to circumvent proper passing of information through function calls) |

## 12.00 Perform linear regression

|  |  |
| --- | --- |
| Learning Objective | Evidence |
| 12.01 State the purpose of regression |  |
| 12.02 Compute and present in equation form the coefficients of a best-fit linear model using visual approximation and the two-point method | Given a plot of raw x-y data, draw a best-fit linear model  Use two points on the drawn line to compute the linear model coefficients: slope (a) and intercept (b)  Presentation of the slope (a) and intercept (b) in the linear model equation |
| 12.03 Manually compute the SSE | Use the dependent variable (y) data and the predicted values of the dependent variable to compute the sum or squares of error |
| 12.04 Manually compute the SST | Use the dependent variable (y) data and the mean of the dependent variable data to compute the sum or squares of deviation |
| 12.05 Manually compute the r-squared value from SSE and SST | Recognize that the result falls between 0 and 1 inclusive |
| 12.06 Add a trendline to a scatter plot of raw x-y data (Excel) | Create a scatter plot of raw bivariate data in Excel  Use Excel chart tools to add a linear trendline to the plot |
| 12.07 Display the equation and r-squared value of a trendline added to a scatter plot (Excel) | Create a scatter plot of raw bivariate data in Excel  Use Excel chart tools to add a linear trendline to the plot  Use Excel chart tools to display the equation on the plot  Revise the equation so that the x and y variables are descriptive  Use Excel chart tools to display the r2 value on the plot |
| 12.08 Manually compute and present in equation form the coefficients of a best-fit linear model using least-squares method | Correct calculation of the summations needed for the two least squares equations  Correct placement of the summations in the two least squares equations  Simultaneous solution of the two least squares equations for the correct linear model coefficients: slope (a) and intercept (b)  Presentation of the slope (a) and intercept (b) in the linear model equation |
| 12.09 Compute the coefficients of a best-fit linear model using least-squares method (MATLAB) | Correct syntax for polyfit  output1 = polyfit(independent\_ vector,dependent\_vector,order\_of\_polynomial)   * polyfit is the MATLAB built-in function * Three input arguments separated by commas: vector1, vector2, and a scalar * One output variable for the coefficients * For linear models, the order of the polynomial is 1   Correct identification of the independent variable (x) and dependent variable (y) used in polyfit |
| 12.10 Compute predicted values using the best-fit linear model (MATLAB) | Correct syntax for polyval  dependent\_predicted\_array = polyval(model\_coeff\_vector,independent\_array)   * polyval is the MATLAB built-in function * Two input arguments separated by commas * One output argument variable assignment * Correct order of input arguments to the polyval   Use appropriate linear model coefficients vector that contains slope and intercept (in that order) in call to polyval  Correctly identified independent variable for the input argument  Appropriate scalar value (for single value predictions) or range of values (e.g., for subsequent plotting) for the independent variable input argument  Correctly identified dependent variable for the output argument |
| 12.11 Plot the best-fit linear regression line on a plot of raw x-y data (MATLAB) | Use the plot command to plot the raw data  Correctly identify the independent and dependent variables when plotting the raw data  Use the plot command to plot the linear model  Correctly identify the independent and dependent variables when plotting the linear model  Use an appropriate method (e.g., hold) to place both the raw data and linear model on the same plot |
| 12.12 Display the results of linear regression (MATLAB) | Display the linear model equation using fprintf  The linear model equation has variable names appropriate to the context of the problem (e.g., does not use variable names x, y)  Correctly identify which polyfit output is the linear model slope  Correctly identify which polyfit output is the linear model intercept  Display the SSE value using fprintf  Display the SST value using fprintf  Display the r-squared value using fprintf  Manage decimal places appropriately |
| 12.13 Define, explain the use of, and relate SSE, SST, r-squared | SSE is a measure of the squared difference between the dependent raw data and predicted values (fit to the data)  SST is a measure of the squared difference between the dependent raw data values and the mean of these values (variability in the data)  r2 is a measure of the extent to which a model explains the variation that exists in the data  Explain how changes in the data set impact SSE, SST, and r2 |
| 12.14 Interpret the slope and intercept of a best-fit linear model | The slope is the rate of change and has units (dependent variable units/independent variable units)  The intercept is the value of the dependent variable when the independent variable equals zero and has units of the dependent variable  Recognize that the intercept may have no meaning with regards to the context of the data |
| 12.15 Interpret the r-squared value | r2 is a measure of the extent to which a model explains the variation that exists in the data  An r2 closer to 1 means that the model does a good job of explaining the variation that exists in the data  An r2 closer to 0 means that the model does a poor job of explaining the variation that exists in the data  Depending on the context of the data, a low r2 might indicate a linear model does a good job of explaining the variation that exists in the data |
| 12.16 Compare data sets based on their best fit linear models and r-squared values | Comparison based on slope  Comparison based on intercept  Comparison based on the extent to which a linear model explains the variation that exists in the data (r2) |
| 12.17 Use the best-fit linear model to make predictions only when appropriate | Independent variable values within the range of the original data set (domain of the model) can be used to make predictions  Independent variable values outside the range of the original data set (domain of the model) must be acknowledged or justified when making predictions  Predicted numerical values must be consistent with the linear model used to make the prediction  Presentation of numerical predictions with appropriate units  Management of the decimal places of numerical predictions |

## 13.00 Perform function discovery and data transformations

| Learning Objective | Evidence |
| --- | --- |
| 13.01 Estimate log10x for any x | Sketch a pair of linear and log number lines representing a range of one order of magnitude  Label the tick marks on the pair of linear and log number lines  Place x on the log number line  Read across from the log number line to the linear line to estimate the log10x value to the nearest tens decimal place |
| 13.02 Identify function types from graphs of bivariate data, specifically linear, power, exponential, and logarithmic | Use intercept, behavior at or near the origin, and asymptotic behavior to identify the function type   * Linear: y = mx + b: has a y intercept and data falls approximately on a line * Exponential: y = b10mx: has a y intercept   + for m>0, as x increases, y increases (concave up)   + for m<0, as x increases, y decreases and asymptotically approaches y = 0 * Power: y = bxm   + for m>0, passes through the origin, as x increases, y increases (concave up for m>1, concave down for 0<m<1)   + for m< 0, there is no intercept; as x approaches 0, y asymptotically approaches x = 0; and as x increases - y asymptotically approaches y = 0 * Logarithmic: x = b10my: has an x intercept; as x approaches 0, y approaches negative infinity; as x increases, y increases (concave down) |
| 13.03 Confirm function identification using a combination of linear and log transformations of the independent and dependent data variables | Identify the independent and dependent data variables that need transformation (or log scaling) to linearize the data  Identify the function type that correspond to the transformations (or log scaling) needed to linearize the data |
| 13.04 Create plots with linear and/or log axis scales (by-hand) | Plots of data using different axis scales to show relationships useful for function discovery   * Linear scale: linear scale on x-axis, linear scale on y-axis * Log-linear scale: log scale on x-axis, linear scale on y-axis * Linear-log scale: linear scale on x-axis, log scale on y-axis * Log-log scale: log scale on x-axis, log scale on y-axis   Data points are plotted correctly on any given graph  Function discovery plots display original independent and dependent data (i.e., non-linearized data) whose relationship is being examined  Each plot has x- and y-axis labels that reference the data in the plot and do not reference the type of scale used |
| 13.05 Create plots with linear and/or log axis scales (Excel) | Plots of data using different axis scales to show relationships useful for function discovery   * Linear scale: linear scale on x-axis, linear scale on y-axis * Log-linear scale: log scale on x-axis, linear scale on y-axis * Linear-log scale: linear scale on x-axis, log scale on y-axis * Log-log scale: log scale on x-axis, log scale on y-axis   Function discovery plots display original independent and dependent data (i.e., non-linearized data) whose relationship is being examined  Each plot has x- and y-axis labels that reference the data in the plot and do not reference the type of scale used  Show the minor gridlines on log scaled axes  Manage the horizontal axis crosses option so that the x-axis tick labels are at the bottom of the plot  Manage the decimal places shown on the x and y axis tick marks |
| 13.06 Create plots with linear and/or log axis scales (MATLAB) | Plots of data with different axis scales to show relationships useful for function discovery are generated using the correct syntax for plotting on different scales   * Linear scale plot: plot command used for linear scale on x-axis, linear scale on y-axis * Log-linear scale plot: semilogx command used for log scale on x-axis, linear scale on y-axis * Linear-log scale plot: semilogy command used for linear scale on x-axis, log scale on y-axis * Log-log scale plot: loglog command used for log scale on x-axis, log scale on y-axis   Function discovery plots display original independent and dependent data (i.e., non-linearized data) whose relationship is being examined  Each plot has x- and y-axis labels that reference the data in the plot and do not reference the type of scale used |
| 13.07 Linearize and plot data appropriately | Linearize the independent variable data correctly based on the diagnosed function type   * Linear: no change to data * Logarithmic: log of independent data * Exponential: no change to independent data * Power: log of independent data   Linearize the dependent variable data correctly based on the diagnosed function type   * Linear: no change to data * Logarithmic: no change to dependent data * Exponential: log of dependent data * Power: log of dependent data   Axes labels (description and units) are correct based on the plotted data |
| 13.08 Linearize a power, exponential, and logarithmic functions | Take the log of both sides of the general form and arrange the terms in the linear form of the equation: Y = MX + B   * Linear: y = mx + b - the linear and general forms are the same * Exponential: y = b10mx becomes log(y) = mx + log(b) * Power: y = bxm  becomes log(y) = mlog(x) + log(b) * Logarithmic: x = b10my becomes y = (1/m)log(x) – (1/m)log(b) |
| 13.09 Determine the linear and general forms of the equations for linear, power, exponential, and logarithmic functions | Identify slope (M) and intercept (B) coefficients for the best-fit linear model of the linearized data   * Linear: use x and y data * Exponential: use x and log(y) transformed data * Power: use log(x) and log(y) transformed data * Logarithmic: use log(x) and y transformed data   Place M and B correctly within the linear form of the equation  Correctly determine the general form constant m from the linear form slope M   * Linear: M = m * Exponential: M = m * Power: M = m * Logarithmic: M = 1/m   Correctly determine the general form constant b from the linear form intercept B   * Linear: B = b * Exponential: B = log(b) * Power: B = log(b) * Logarithmic: B = 1/m\*log(b)   Replace (m) correctly within the general form of the equation   * Linear: y = mx +b * Exponential: y = b10mx * Power: y = bxm * Logarithmic: x = b10my   Replace (b) correctly within the general form of the equation   * Linear: y = mx +b * Exponential: y = b10mx * Power: y = bxm * Logarithmic: x = b10my |
| 13.11 Use the function to make predictions only when appropriate | Independent variable values within the range of the original data set (domain of the function model) can be used to make predictions  Independent variable values outside the range of the original data set (domain of the function model) must be acknowledged or justified when making predictions  Predicted numerical values must be consistent with the equation used to make the prediction  Presentation of numerical predictions with appropriate units  Management of the decimal places of numerical predictions |