Exam 2: Learning Objectives under Evaluation

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

Exam 2: Learning Objectives under Evaluation

Learning Objective	Evidence
	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.

Exam 2: Learning Objectives under Evaluation

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

Exam 2: Learning Objectives under Evaluation

<table>
<tr><th>Learning Objective</th><th>Evidence</th></tr>
<tr><td></td><td>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</td></tr>
<tr><td>11.05 Match the variables names used in the function definition line to those used in the function code</td><td>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</td></tr>
<tr><td>11.06 Execute a user-defined function</td><td>Correct syntax to call a function:
<code>[output1,…,outputN] = function_name(input1,…,inputM)</code>
Call does not contain keyword <code>function</code>
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</td></tr>
</table>

Exam 2: Learning Objectives under Evaluation

Learning Objective	Evidence
	• 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)

Exam 2: Learning Objectives under Evaluation

12.00 Perform linear regression

<table>
<tr><th>Learning Objective</th><th>Evidence</th></tr>
<tr><td>13.01 Estimate log10x for any x</td><td> 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 </td></tr>
<tr><td>13.02 Identify function types from graphs of bivariate data, specifically linear, power, exponential, and logarithmic</td><td> 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)</td></tr>
<tr><td>13.03 Confirm function identification using a combination of linear and log transformations of the independent and dependent data variables</td><td> 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 </td></tr>
<tr><td>13.04 Create plots with linear and/or log axis scales (by-hand)</td><td> 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</td></tr>
</table>

Exam 2: Learning Objectives under Evaluation

Learning Objective	Evidence
	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

Exam 2: Learning Objectives under Evaluation

Learning Objective	Evidence
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: $y = b10^{mx}$ becomes log(y) = mx + log(b) • Power: $y = bx^{m}$ becomes log(y) = mlog(x) + log(b) • Logarithmic: $x = b10^{my}$ 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

Exam 2: Learning Objectives under Evaluation

Learning Objective	Evidence
	• 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: $y = b10^{mx}$ • Power: $y = bx^{m}$ • Logarithmic: $x = b10^{my}$ Replace (b) correctly within the general form of the equation • Linear: $y = mx + b$ • Exponential: $y = b10^{mx}$ • Power: $y = bx^{m}$ • Logarithmic: $x = b10^{my}$
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

Exam 2: Learning Objectives under Evaluation