Data Representation in MATLAB: From Raw Data to Insight

Background

In the modern computational landscape, scientists and engineers work with diverse data types—from numerical spreadsheets and audio recordings to digital images. The ability to seamlessly import, validate, and analyze this heterogeneous data is a foundational skill. This practical exercise guides you through the complete data handling pipeline in MATLAB, demonstrating how its core data structure—the array—unifies the representation of numbers, sound, and pictures. You will learn to move from raw data to meaningful insight through systematic inspection and visualization.

Core Concept: Whether it's a number in a cell, a sound wave sample, or a pixel's color, in MATLAB, it's all an array.

Part 1: Project Setup & Data Acquisition

Task - Setting Up Your Workspace

All good data analysis begins with organization. Your first task is to create a dedicated folder for this project and set it as MATLAB's Current Folder.

`matlab code`

```
% Define the full path to a new folder on your Desktop
currentFolder = fullfile(getenv('USERPROFILE'), 'Desktop', 'CIEPracticals','Prac1_DataLab');

% Create the folder if it doesn't exist
if ~exist(currentFolder, 'dir')
    mkdir(currentFolder);
end
% Change the current directory to your new project folder
cd(currentFolder);
```

Question: Why is it important to set a Current Folder in MATLAB?

Part 2: Working with Spreadsheet Data

Spreadsheets are a common format for storing structured, tabular data. We will import this data and see how it can be represented and analyzed as arrays and tables.

Task 2.1 - Importing Spreadsheet Data

- Locate the provided `ElectricityData.xlsx` file.
- Copy and paste it into your current folder.
- Import the data into the MATLAB workspace using two different functions to see the difference.

```
% Import as a numeric matrix (ignores text headers
electricityMatrix = readmatrix('electricityData.xlsx');

% Import as a table (preserves headers and data types)
electricityTable = readtable('electricityData.xlsx');
```

Task 2.2 - Validation & Inspection

Before analysis, we must check the data's integrity and structure.

`matlab code`

```
% Inspection Commands

% Get dimensions of the matrix
matrixSize = size(electricityMatrix);

% Get dimensions of the table
tableSize = size(electricityTable);

% Get a detailed summary of the table
summary(electricityTable)

% Check data types per column
dataTypes = varfun(@class, electricityTable, 'OutputFormat', 'cell');

% Check for missing values
missingValues = ismissing(electricityTable);
```

Inspection Questions:

1. How many rows and columns of data were imported?
2. What is the fundamental difference between `electricityMatrix` and `electricityTable`?
3. Are there any missing values in the dataset?

Task 2.3 - Visualization

Visualization helps us see patterns and relationships in numerical data.

`matlab code`

```
% Create a plot of the total revenue by year
figure;
plot(electricityTable.Date, electricityTable.Total);
title('Total Revenue by year');
xlabel('Year');
ylabel('Total Revenue ($)');
```

```
```

```
% Create a box plot to see the distribution of energy consumption in each sector
figure;
boxplot(electricityTable{:, 2:5}, 'Labels', electricityTable.Properties.VariableNames(2:5));
title('Distribution of Electricity Consumption Across Sectors');
ylabel('Energy Consumed (kWh)');
```

Interpretation Questions:

1. In which year was the highest revenue generated?
2. Looking at the box plot, which sector shows the most consistent energy consumption?

Part 3: Working with Audio Data

Audio is a one-dimensional signal representing how air pressure changes over time. In MATLAB, it becomes a single column vector (mono) or a two-column matrix (stereo).

Task 3.1 - Recording and Importing Audio

- Use your computer's voice recorder app to record a clear 3-5 second clip of your voice (e.g., saying "MATLAB is powerful!").
- Save the file as `myVoice.wav` in your project folder.
- Import the audio into MATLAB.

`matlab code`

```
% Import the audio file
% myVoice: the sampled audio data (the amplitude values)
% Fs: the sample rate (samples per second)
[myVoice, Fs] = audioread('myVoice.wav');
```

Task 3.2 - Validation & Inspection

Let's understand the properties of our audio signal.

`matlab code`

```
% Inspection Commands
audioLength = length(myVoice)    % Total number of samples
audioDuration = audioLength / Fs % Duration of the audio in seconds
audioSampleRate = Fs             % Sampling frequency (Hz)
audioDataType = class(myVoice)   % Data type of the amplitude values

% Check for clipping (distortion caused by amplitude hitting max)
maxAmplitude = max(abs(myVoice))
```

Inspection Questions:

1. How long is your audio clip in seconds?
2. What is the sample rate? What is the practical implication of this number?
3. Is there any evidence of clipping in your recording (i.e., are any absolute amplitudes very close to 1.0)?

Task 3.3 - Visualization

Visualizing audio helps us "see" the sound.

matlab code

```
% Create a time vector for the x-axis
timeVector = (0:length(myVoice)-1) / Fs;

% Plot the audio waveform
figure;
subplot(2,1,1);
plot(timeVector, myVoice);
title('Waveform of My Voice Recording');
xlabel('Time (seconds)');
ylabel('Amplitude');
grid on;

% Plot a histogram of the amplitude values
subplot(2,1,2);
histogram(myVoice, 50);
title('Distribution of Audio Amplitudes');
xlabel('Amplitude');
ylabel('Frequency');
```

Interpretation Questions:

- Does your waveform show a repeating pattern corresponding to the syllables of your speech?
- Is the amplitude histogram centered around zero and roughly symmetrical?

Part 4: Working with Image Data

A digital image is essentially a grid of colored points called pixels. A grayscale image is a 2D array where each value is a brightness. A color image is a 3D array (Height x Width x 3), where the third dimension holds the Red, Green, and Blue color channels.

Task 4.1 - Capturing and Importing an Image

1. Use your phone or webcam to take a clear, well-lit picture of a single object.
2. Transfer and save the file as `myImage.jpg` in your project folder.
3. Import the image into MATLAB.

matlab code

```
% Import the image file
myImage = imread('myImage.jpg');
```

Task 4.2 - Validation & Inspection

Let's dissect the structure of our image data.

matlab code

```
% Inspection Commands
imageDimensions = size(myImage)  % Get the dimensions [rows, cols, channels]
imageDataType = class(myImage)   % Get the data type (e.g., uint8)

% Get basic statistics about pixel intensities
minPixel = min(myImage(:));
maxPixel = max(myImage(:));
meanPixel = mean(double(myImage(:))); % Convert to double for calculation
```

Inspection Questions:

1. Is your image color (3 dimensions) or grayscale (2 dimensions)?
2. What is the data type and range of the pixel values? (e.g., `uint8` means integers from 0 to 255).
3. What are the minimum, maximum, and average pixel intensity values in your image?

Task 4.3 - Visualization

Visualizing the image itself is primary, but visualizing its pixel distribution reveals contrast and composition.

matlab code

```
% Display the image
figure;
subplot(1,2,1);
imshow(myImage);
title('My Imported Image');

% Display the histogram of pixel intensities
subplot(1,2,2);
if ndims(myImage) == 3
    % For color images, let's look at the grayscale histogram
    imhist(rgb2gray(myImage));
else
    % For grayscale images
    imhist(myImage);
end

title('Pixel Intensity Histogram');
xlabel('Intensity Value');
ylabel('Number of Pixels');
```

Interpretation Questions:

1. Does the histogram of your image show a good range of contrast (a spread-out distribution)?
2. Where are the pixel values concentrated? In the darks, mid-tones, or highlights? What does this say about your image's exposure?

Part 5: Final Synthesis & Report

Task 5.1 - Unified Workspace Review

In your MATLAB Workspace, you should now have variables for all three data types: `electricityTable` (spreadsheet), `myVoice` and `Fs` (audio), and `myImage` (image).

Final Reflection:

Despite their different origins and natures, all this data is now represented as arrays in your workspace. Briefly describe the "shape" and meaning of the data in each of these core variables.

Task 5.2 - Submission

Create a single MATLAB Live Script that executes all the tasks from this document. Your script should be well-commented and include:

1. **Code**: All the commands for setup, import, inspection, and visualization.
2. **Answers**: Correct answers to inspection/interpretation questions are provided as text in the Live Script
3. **Outputs**: The generated figures and answers to the inspection/interpretation questions embedded as text.
4. **Summary**: A final section (approx. 200 words) titled "Data Unification in MATLAB," discussing how the array structure serves as a universal data representation and reflecting on the insights gained from the validation and visualization steps for each data type.

Answer Key for Lab Questions

Part 1: Project Setup

- Question: Why is it important to set a Current Folder in MATLAB?
- Answer: It ensures MATLAB looks for files (like data and scripts) in the correct location and saves any new files to an organized, predictable directory. This prevents path errors and keeps the project tidy.

Part 2: Spreadsheet Data

Inspection Questions:

a) How many rows and columns of data were imported?

- Answer: 315 rows and 5 columns (for the table).

b) What is the fundamental difference between `electricityMatrix` and `electricityTable`?

- Answer: `electricityMatrix` is a numeric-only array, so it loses the text headers and the 'Region' column. `electricityTable` preserves the column headers as variable names and can store mixed data types (text and numbers).

c) Are there any missing values in the dataset?

- Answer: yes. The `summary(electricityTable)` command return a total of 39 missing values.

Interpretation Questions:

a) In which year was the highest revenue generated?

- Answer: 2011.

b) Looking at the box plot, which sector shows the most consistent energy consumption across regions?

- Answer: The Industrial sector. This is indicated by the box plot for 'Industrial' having the shortest box and whiskers, meaning the data points are clustered closely together, showing low variability.

Part 3: Audio Data

Inspection Questions:

a) How long is your audio clip in seconds?

- Answer: *Student-specific, calculated as* `audioLength / Fs`. (e.g., "My audio is 4.1 seconds long.")

b) What is the sample rate? What is the practical implication of this number?

- Answer: *Student-specific, from the* `Fs` *variable*. (e.g., "The sample rate is 44100 Hz.") The practical implication is that it defines the quality of the audio; a higher sample rate means the original sound wave was sampled more frequently, leading to a more accurate digital representation.

c) Is there any evidence of clipping?

- Answer: *Student-specific, based on* `max(abs(myVoice))`. (e.g., "No, the maximum amplitude is 0.45, which is not close to 1.0." OR "Yes, the maximum amplitude is 0.99, indicating potential clipping.")

Interpretation Questions:

a) Does your waveform show a repeating pattern corresponding to the syllables of your speech?

- Answer: Yes, the waveform should show clear, repeating oscillations and amplitude changes that correspond to the words and syllables spoken.

b) Is the amplitude histogram centered around zero and roughly symmetrical?

- Answer: Yes, for typical voice audio, the histogram should be roughly bell-shaped and centered near zero, indicating a balanced signal without significant DC offset.

Part 4: Image Data

Inspection Questions:

a) Is your image color (3 dimensions) or grayscale (2 dimensions)?

- Answer: *Student-specific, based on* `size(myImage)`. (e.g., "My image is color; its dimensions are [480 640 3].")

b) What is the data type and range of the pixel values?

- Answer: The data type is typically `uint8`. The range for `uint8` is integers from 0 (black) to 255 (white).

c) What is the minimum, maximum, and average pixel intensity values in your image?

- Answer: *Student-specific, based on* `minPixel`*,* `maxPixel`*, and* `meanPixel`. (e.g., "Min: 15, Max: 243, Mean: 118.7.")

`Interpretation Questions:`

a) Does the histogram of your image show a good range of contrast (a spread-out distribution)?

- Answer: *Student-specific*. (e.g., "Yes, the histogram spans from near 0 to near 255, indicating good contrast." OR "No, the histogram is clustered mostly in the dark tones, so my image is low-contrast and underexposed.")

b) Where are the pixel values concentrated? What does this say about your image's exposure?

- Answer: *Student-specific*. (e.g., "The pixels are concentrated in the mid-tones (around 80-170), which suggests a well-exposed image." OR "The pixels are concentrated in the highlights (170-255), which suggests an overexposed image.")

`Part 5: Final Synthesis`

`Final Reflection:` Briefly describe the "shape" and meaning of the data in each variable.

- `electricityTable`: A 4x6 table. The shape represents 4 observations (regions) and 6 variables. The data is structured, tabular data with mixed types (categorical and numerical).
- `myVoice`: A long column vector (mono) or Nx2 matrix (stereo). The shape represents N amplitude samples taken over time. The data is a one-dimensional time-series signal.
- `myImage`: A 2D matrix (grayscale) or 3D array (color). The shape represents [height, width, color_channels]. The data is a spatial representation of light intensity/color.