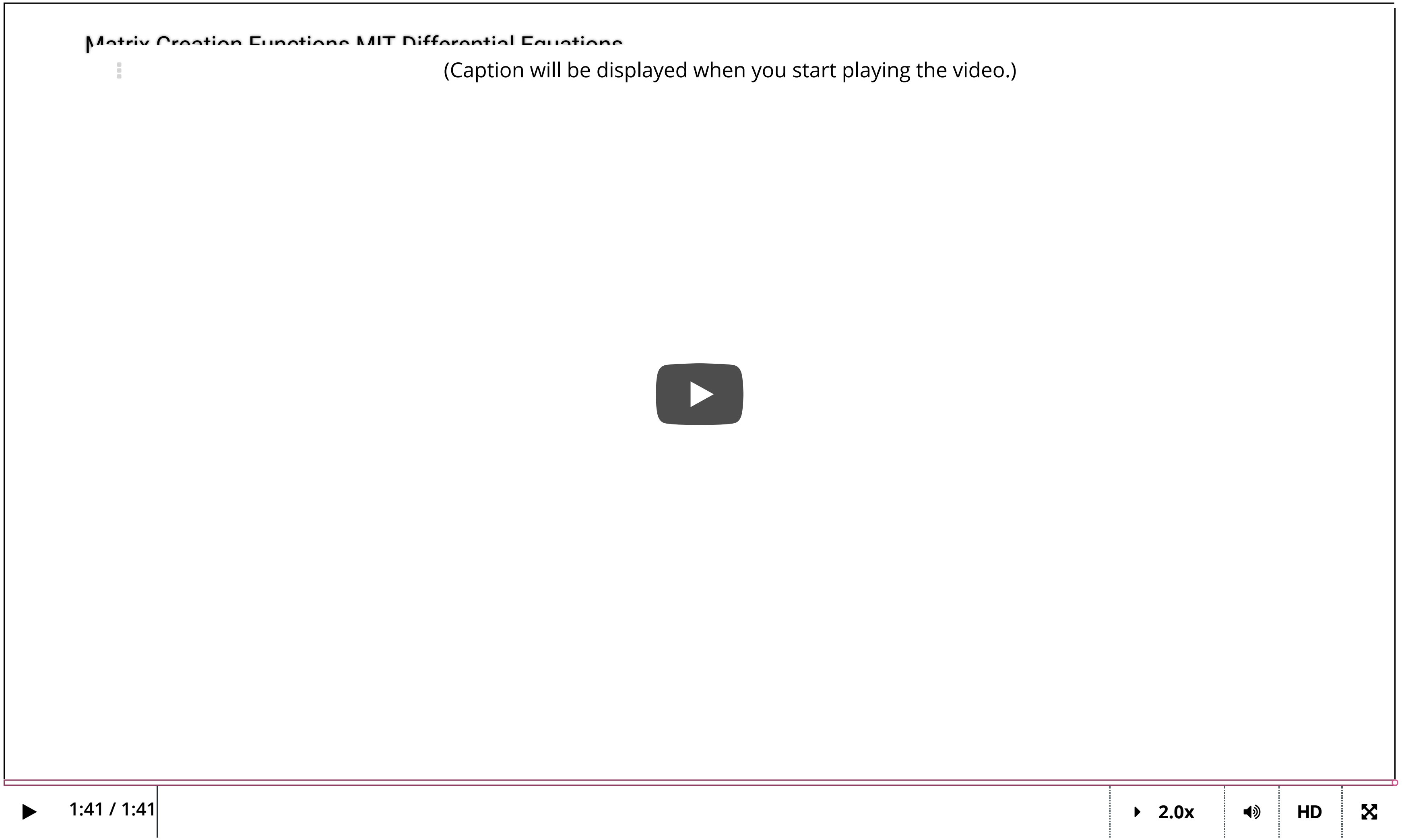
2. Diagonal and superdiagonal matrices

Matrix creation functions



Matrix Creation Functions MIT Differential Equations

(Caption will be displayed when you start playing the video.)

MATLAB and diagonal matrices (External resource) (1.0 points possible)

Diagonal matrices

MATLAB's command diag(_) has two uses

- If $\mathbf{v}$ is a vector of length n, then

```
D = diag(v);
```

creates an $n \times n$ diagonal matrix with the elements of $\mathbf{v}$ along the diagonal

- If $\mathbf{A}$ is an $n \times n$ square matrix, then

```
x = diag(A)
```

creates a column vector $\mathbf{x}$ out of the elements from the main diagonal of $\mathbf{A}$.

Complete the script below, which illustrates the two uses of the diag command.

Script ?

Save   Reset   MATLAB Documentation (https://www.mathworks.com/help/)

```
% First create a random 15x15 matrix A using the rand() command
A = rand(15,15);
% Now create a new variable v which is a column vector
% made from the main diagonal of A using the diag() function
v = diag(A);
% Now create a 10x10 diagonal matrix D made from the first ten elements of v.
% Think carefully about the easiest way to do this. Remember you can create as many new variables as you

D = diag(v(1:10));
```

▶ Run Script ?

Assessment: All Tests Passed

Submit ?

✓ **A correctly defined**

✓ v correctly defined

✓ D correctly defined

Output

```
Code ran without output.
```

MATLAB and superdiagonal matrices (External resource) (1.0 points possible)

Superdiagonal and subdiagonal matrices

Let $\mathbf{v}$ be a vector of length n, then

```
D = diag(v,1);
```

creates an $n+1 \times n+1$ matrix with the elements of $\mathbf{v}$ above the main diagonal

Example 1:

```
v = ones(4,1)*5; %Creates a vector with 4 entries all equal to 5.
A = diag(v,1); %Creates the 5x5 matrix displayed below.
```

$$\mathbf{A} = \begin{pmatrix} 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 5 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

You can create entries along a subdiagonal by using negative numbers rather than positive numbers.

Example 2:

```
v = ones(3,1)*-1; %Creates a vector with 3 entries all equal to -1.
A = diag(v,-2); %Creates the 5x5 matrix displayed below.
```

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{pmatrix}$$

Script ?

Save Reset MATLAB Documentation (https://www.mathworks.com/help/)

```
%Create a column vector v with entries 1,2,3,4 in that order.
v = (1:4)';

%Create a matrix A with the entries of v on the 3rd super diagonal.
A = diag(v,3);

%Create a matrix B with the entries of v on the (-1)st sub diagonal.
B = diag(v,-1);
```

▶ Run Script ?

Assessment: All Tests Passed

Run Pretest ? Submit ?

✓ **Is v correct?** (Pretest)

✓ **Is A correct?**

✓ **Is B correct?**

Output

```
Code ran without output.
```

MATLAB and tridiagonal matrices (External resource) (1.0 points possible)

Tridiagonal matrices

MATLAB has a way of creating tridiagonal matrices (matrices with entries only on the diagonal, and super and sub diagonals).

These are saved as "sparse" matrices which are a data type that only saves the locations of the nonzero entries.

For example, we can create a 10x10 matrix $\mathbf{A}$ with entries 1 along the main subdiagonal, 2 along the diagonal, and 4 along the super (
Attempting to find the entry in the (1,1) position will give the following error.

```
n=10;
A = gallery('tridiag',n,1,2,4);
```

However, to get a matrix of the data type that we are used to, you only need add the command full() as follows.

```
n=10;
A = full(gallery('tridiag',n,1,2,4));
```

Try creating these matrices but smaller, 5x5 instead of 10x10 in the MATLAB window below without the semicolon at the end to see ho
that the grader tests data type, so a sparse matrix and a full matrix are not seen as equivalent.

Script ?

Save Reset MATLAB Documentation (https://www.mathworks.com/help/)

```
% Create a 5x5 tridiagonal matrix A with entries 1, 2, 4 as a sparse matrix.
A = gallery('tridiag', 5, 1, 2, 4);

% Create a 5x5 tridiagonal matrix B with entries 1, 2, 4 as a *full* matrix.
B = full(gallery('tridiag', 5, 1, 2, 4));
```

▶ Run Script ?

Assessment: All Tests Passed

Submit ?

✓ **A is correct?**

✓ **B is correct?**

Output

```
Code ran without output.
```

2. Diagonal and superdiagonal matrices

Hide Discussion

Topic: Unit 2: Boundary value problems and PDEs / 2. Diagonal and superdiagonal matrices

Add a Post

Show all posts ⌄   by recent activity ⌄

☑ Page working extremely slowly 3