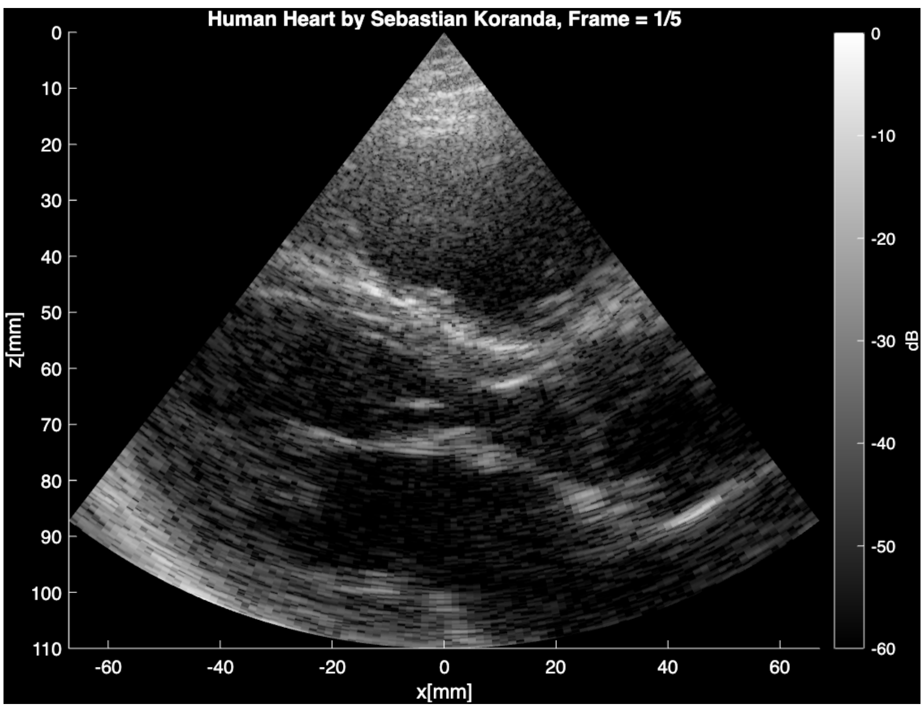
Module 1

Sebastian Koranda

September 2025

Exercise 1



Exercise 2

1a)

```
x_row = 0 : 2 : 6
```

gives row vector of numbers 0 to 6 with 2 step increments

$$\boldsymbol{x}_{row} = \begin{bmatrix} 0 & 2 & 4 & 6 \end{bmatrix}$$

```
x_col = x_row.'
```

gives transposed column vector

$$\boldsymbol{x}_{col} = \begin{bmatrix} 0 \\ 2 \\ 4 \\ 6 \end{bmatrix}$$

```
x1 = [1,2,3]
```

gives row vector

$$\boldsymbol{x}_1 = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$

```
x2 = [1;2;3]
```

gives column vector

$$\boldsymbol{x}_2 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

```
x3 = linspace(1,10,4)
```

gives row vector of 4 evenly spaced numbers from 1 to 10

$$\boldsymbol{x}_3 = \begin{bmatrix} 1 & 4 & 7 & 10 \end{bmatrix}$$

1b)

```
a = x_row * x_col
```

inner product

$$a = \boldsymbol{x}_{row}\boldsymbol{x}_{col} = \begin{bmatrix} 0 & 2 & 4 & 6 \end{bmatrix} \begin{bmatrix} 0 \\ 2 \\ 4 \\ 6 \end{bmatrix} = 56$$

```
b = x_col * x_row
```

inner product

$$\boldsymbol{B} = \boldsymbol{x}_{col}\boldsymbol{x}_{row} = \begin{bmatrix} 0 \\ 2 \\ 4 \\ 6 \end{bmatrix} \begin{bmatrix} 0 & 2 & 4 & 6 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 4 & 8 & 12 \\ 0 & 8 & 16 & 24 \\ 0 & 12 & 24 & 36 \end{bmatrix}$$

```
c = x_row .* x_col
```

`x_row` is treated as a 4×4 matrix with 4 identical rows, and `x_col` as a 4×4 matrix with 4 identical columns, then element-wise (Hadamard) product

$$\boldsymbol{C} = \begin{bmatrix} 0 & 2 & 4 & 6 \\ 0 & 2 & 4 & 6 \\ 0 & 2 & 4 & 6 \\ 0 & 2 & 4 & 6 \end{bmatrix} \odot \begin{bmatrix} 0 & 0 & 0 & 0 \\ 2 & 2 & 2 & 2 \\ 4 & 4 & 4 & 4 \\ 6 & 6 & 6 & 6 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 4 & 8 & 12 \\ 0 & 8 & 16 & 24 \\ 0 & 12 & 24 & 36 \end{bmatrix}$$

```
d = x_row .* x_col'
```

element-wise (Hadamard) product

$$\boldsymbol{d} = \boldsymbol{x}_{row} \odot \boldsymbol{x}_{col}^T = \begin{bmatrix} 0 & 2 & 4 & 6 \end{bmatrix} \odot \begin{bmatrix} 0 & 2 & 4 & 6 \end{bmatrix} = \begin{bmatrix} 0 & 4 & 16 & 36 \end{bmatrix}$$

```
e = x_row' .* x_col
```

element-wise (Hadamard) product

$$\boldsymbol{e} = \boldsymbol{x}_{row}^T \odot \boldsymbol{x}_{col} = \begin{bmatrix} 0 \\ 2 \\ 4 \\ 6 \end{bmatrix} \odot \begin{bmatrix} 0 \\ 2 \\ 4 \\ 6 \end{bmatrix} = \begin{bmatrix} 0 \\ 4 \\ 16 \\ 36 \end{bmatrix}$$

1c)

```
z_row = x_row + 2j*x_row
```

$$\begin{aligned}\boldsymbol{z}_{row} = \boldsymbol{x}_{row} + 2i\boldsymbol{x}_{row} &= \begin{bmatrix}0 & 2 & 4 & 6\end{bmatrix} + 2i\begin{bmatrix}0 & 2 & 4 & 6\end{bmatrix} \\ &= \begin{bmatrix}0 & 2 & 4 & 6\end{bmatrix} + \begin{bmatrix}0+0i & 0+4i & 0+8i & 0+12i\end{bmatrix} \\ &= \begin{bmatrix}0+0i & 2+4i & 4+8i & 6+12i\end{bmatrix} \quad (1)\end{aligned}$$

1d)

```
z_col1 = z_row'
```

$$\boldsymbol{z}_{col1}^T = \begin{bmatrix}0+0i \\ 2-4i \\ 4-8i \\ 6-12i\end{bmatrix}$$

```
z_col2 = z_row.'
```

$$\boldsymbol{z}_{col2}^* = \begin{bmatrix}0+0i \\ 2+4i \\ 4+8i \\ 6+12i\end{bmatrix}$$

2a)

```
A = nan(3,3);
A(1,:) = [1,1,1];
A(2,:) = [2,2,2];
A(3,:) = [3,3,3];
```

$$A = \begin{bmatrix}1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3\end{bmatrix}$$

```
B = [1 1 1 1 1; 2 2 2 2 2; 3 3 3 3 3; 4 4 4 4 4; 5 5 5 5 5]
```

$$B = \begin{bmatrix}1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 & 3 \\ 4 & 4 & 4 & 4 & 4 \\ 5 & 5 & 5 & 5 & 5\end{bmatrix}$$

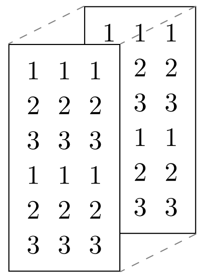
```
C = [1 2 3
     4 5 6]
```

$$C = \begin{bmatrix}1 & 2 & 3 \\ 4 & 5 & 6\end{bmatrix}$$

2b)

```
B = repmat(A,[2 1 2]);
```

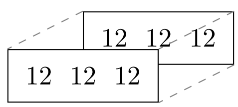
Repeat A 2 times in the first dimension, once in the second dimension, and 2 times in the third dimension:



2c)

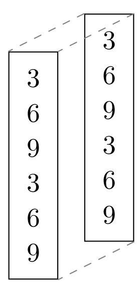
```
sum1 = sum(B);
```

Sum of elements of `B` along the first array dimension whose size does not equal to 1:




```
sum2 = sum(B,2);
```

Sum of elements of `B` along dimension 2:



```
sum3 = sum(B,3);
```

Sum of elements of `B` along dimension 3:

2	2	2
4	4	4
6	6	6
2	2	2
4	4	4
6	6	6

```
sum4 = sum(B, 'all');
```

Sum of elements of `B` along all dimensions:

72

```
sum5 = sum(sum(sum(B)));
```

Iteratively sum elements of `B` along the first dimension, 3 times. Effectively same as previous case:

72

2c)

`sum1` has size $1 \times 3 \times 2$. The first dimension of length 1 is a singleton dimension. The `squeeze`-function removes every singleton dimension, giving us a 3×2 matrix.

```
sum1_squeezed = squeeze(sum1);
```

12	12
12	12
12	12