1. **QR factorization**

   (a) Create a function `qr2` that takes as input two matrices A and B and outputs the (thin) QR factorization of the matrix

   $$\begin{pmatrix} A \\ B \end{pmatrix}.$$

   Use the `hstack` function. What is the output of the following code snippet?

```
A = np.ones(20, 5)
B = np.diag(np.arange(1, 6))
Q, R = qr2(A, B)
print R
```

   (b) Write a function `qr3` that takes as input two matrices A and B and does the following:

      i. Computes the QR factorization of A: $A = Q_A R_A$.
      ii. Computes the QR factorization of B: $A = Q_B R_B$.
      iii. Computes the QR factorization of $C = \begin{pmatrix} R_A \\ R_B \end{pmatrix}$: $C = Q_C R_C$.
      iv. Outputs the tuple (Q_A, Q_B, Q_C, R_C)

   What is the output of the following code snippet?

```
A = np.ones(20, 5)
B = np.diag(np.arange(1, 6))
QA, QB, QC, RC = qr3(A, B)
print RC
```

   How does this compare to the output in part (a)?

   (c) Using the outputs from part (b), compute the matrix

   $$Q_D = \begin{pmatrix} Q_A & 0_{m_1 \times n} \\ 0_{m_2 \times n} & Q_B \end{pmatrix} Q_C$$

   $0_{m_1 \times n}$ means a matrix of all zeros consisting of m_1 rows and n columns, where m_1 is the number of rows of Q_A and n is the number of columns of Q_B.

   How does Q_D compare to the output Q from part (a)?

2. **Low-rank approximations with the SVD**

(a) Write a function `laplace` that takes as input a vector v of length n and outputs a matrix K of size $n \times n$ such that

$$K_{ij} = \begin{cases} \frac{1}{|v_i - v_j|} & i \neq j \\ 0 & i = j \end{cases}$$

When the elements of v are points in three-dimensional space, $1/\|v_i - v_j\|_2$ is called the Laplace kernel.

(b) Define v by `v = np.arange(0.1, 10, 0.5)`. v should have length 20. Define K by `K = laplace(v)` and let

$$K_1 = K[0 : 10, 10 : 20],$$

i.e., K_1 is the 10×10 upper-right block of K. Plot the singular values of K_1. What do you notice?

(c) Consider the matrix K_1 from part (b) and its singular value decomposition

$$K_1 = U \Sigma V^T.$$

Compute the matrix K_2, defined by

$$K_2 = U[:, 0 : 3] \Sigma[0 : 3, 0 : 3] V^T[0 : 3, :].$$

In other words, K_2 is the product of the first three columns of U, the upper-left 3×3 block of Σ, and the first three rows of V^T. (Remember that we defined Σ as a matrix, but `np.linalg.svd` returns just a vector of values since Σ is diagonal). K_2 is called a *low-rank approximation* of K_1.

Define the vector x by `x = np.ones(10)`. Compute the vector

$$K_1 x - K_2 x.$$

Explain the output.