GPU matrix-vector multiplication

Note: Using dynamically allocated two-dimensional arrays for matrices in CUDA is not as efficient as using one-dimensional arrays with your own 2D indexing,- For this exercise, you are encouraged to use one-dimensional arrays.

Exercise 4:

Write a CUDA C program for matrix-vector multiplication on the GPU - you can use the code from last week as a starting point.

1. Write a 'naive' version, where each thread computes one element of the output vector.
2. Compare the kernel runtime to your fastest OpenMP version. Did you get any speed-up? Make sure that the speed-up calculation is fair.
3. Compare with the `DGEMV` function for GPUs provided by Nvidia's CUBLAS library.
4. Make a version that can run simultaneously on two GPUs by splitting the task equally between them (you may assume that the matrix size is an even number).