LU decomposition using CUDA

In this example we will create a random 4x5 matrix using uniform distribution and calculate its LU decomposition matrix. The calculation will be achieved using the Nvidia GPU card and CUDA with a group of MatDeck functions that incorporate ArrayFire functionalities.

First, we will set the environment to use the GPU for calculations. Using the function, afp_supported_backends, a list of all supported backends that can be used for calculations will be produced. In our case, calculations can be made on the CPU, using OpenCL or CUDA framework.

$$\text{afp\_supported\_backends}() = \begin{bmatrix} \text{"cpu"} \\ \text{"opencl"} \\ \text{"cuda"} \end{bmatrix}$$

Default environment for calculations is the CPU. We can change the current environment with the function, afp_set_backend, and check which environment is currently in use with the afp_backend function.

$$\text{afp\_set\_backend}(\text{"cuda"}) = \text{true}$$

$$\text{afp\_backend}() = \text{"cuda"}$$

In each environment, there can be several devices which support calculations within it. To check number of devices which supports calculations in the current environment, use the function, afp_get_device_count, and the functions afp_get_device and afp_set_device to check/change current device.

$$\text{afp\_get\_device\_count}() = 1$$

$$\text{afp\_get\_device}() = 0$$

$$\text{afp\_set\_device}(0) = \text{true}$$

To display information about currently selected device, use the function afp_device_info

$$\text{afp\_device\_info}() = \begin{bmatrix} \text{"NVIDIA\_GeForce\_940MX"} \\ \text{"CUDA"} \\ \text{"v11.2"} \\ \text{"5.0"} \end{bmatrix}$$

Finally, we have set CUDA as a calculation backend and set the device with number 0 - Nvidia GeForce graphic card as a device on which we will do all calculations.

Let's create a uniformly random 4x5 matrix with real values.

$$A := \text{afp\_randu}(4, 5, \text{"real"})$$

We can print variable A to check that the input matrix is generated.

$$A = \begin{bmatrix} 0.918 & 0.362 & 0.148 & 0.829 & 0.614 \\ 0.711 & 0.481 & 0.061 & 0.198 & 0.736 \\ 0.813 & 0.140 & 0.951 & 0.036 & 0.839 \\ 0.13 & 0.218 & 0.468 & 0.312 & 0.252 \end{bmatrix}$$

Now, we can do LU decomposition calculations on matrix A and place the resulting vector in variable B. Resulting vector contains lower triangle matrix L, upper triangle matrix U and pivot vector.

$$B := \text{afp\_lu}(A)$$

$$B = \begin{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0.774 & 1 & 0 & 0 \\ 0.885 & -0.899 & 1 & 0 \\ 0.141 & 0.833 & 0.639 & 1 \end{bmatrix} & \begin{bmatrix} 0.918 & 0.362 & 0.148 & 0.829 & 0.614 \\ 0 & 0.200 & -0.054 & -0.444 & 0.260 \\ 0 & 0 & 0.771 & -1.096 & 0.529 \\ 0 & 0 & 0 & 1.264 & -0.39 \end{bmatrix} \\ \ldots \end{bmatrix}$$

There are separate functions for every member of the resulting vector.
Function, afp_lu_low will, calculate the lower triangle matrix L

$$\text{afp\_lu\_low}(A) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0.774 & 1 & 0 & 0 \\ 0.885 & -0.899 & 1 & 0 \\ 0.141 & 0.833 & 0.639 & 1 \end{bmatrix}$$

Function afp_lu_upp will calculate upper triangular matrix U

$$\text{afp\_lu\_upp}(A) = \begin{bmatrix} 0.918 & 0.362 & 0.148 & 0.829 & 0.614 \\ 0 & 0.200 & -0.054 & -0.444 & 0.260 \\ 0 & 0 & 0.771 & -1.096 & 0.529 \\ 0 & 0 & 0 & 1.264 & -0.39 \end{bmatrix}$$

And finally, function afp_lu_piv will return the pivot vector of the LU decomposition

$$\text{afp\_lu\_piv}(A) = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \end{bmatrix}$$