Otto-von-Guericke-Universität Magdeburg
Max-Planck-Institut for Dynamics of Complex Technical Systeme
Computational Methods in Systems and Control Theory

Dr. Jens Saak, Dipl.-Math. Martin Köhler
Website: http://www.mpi-magdeburg.mpg.de/2896761/sc2

summer term 2015

Scientific Computing 2
Exercise 5
06/25/2015

Exercise 1:

The general matrix-matrix product (GEMM)

$$C \leftarrow \alpha AB + \beta C,$$

where $A \in \mathbb{R}^{n\times m}$, $B \in \mathbb{R}^{m\times p}$, $C \in \mathbb{R}^{n\times p}$ and $\alpha, \beta \in \mathbb{R}$, is one of the most efficiently implemented operations on CPUs and GPUs. In order to accelerate programs using GPUs or similar accelerator devices it is necessary to move such operations onto the accelerator. Develop the following operations as pseudo-code algorithms on top of an accelerator device. Denote where data has to be transfer between host an device and which operations are performed on the host or on the device. Assume that all matrices live in the memory of the CPU when that computation starts and the result should be available there as well after the computation finishes.

a.) The GEMM operation, as shown above, should be moved to the accelerator in a straight forward way. Handle the scalar parameters α and β appropriately to avoid unnecessary data transfers.

b.) The GEMM operation can be rearranged as

$$\begin{bmatrix} C_1 & C_2 & \cdots & C_k \end{bmatrix} \leftarrow \alpha A \begin{bmatrix} B_1 & B_2 & \cdots & B_k \end{bmatrix} + \beta \begin{bmatrix} C_1 & C_2 & \cdots & C_k \end{bmatrix},$$

where B and C are split into columns block of size $m \times q$ or respectively $n \times q$. Develop a pseudo code algorithm which only stores the matrix A completely on the device and uses asynchronous operations to transfer and compute the blocks of B and C. The asynchronous operations should be realized using the streams concept of CUDA. Take care about the scalars α and β as well.

Hint: You can assume that all BLAS-like linear algebra operation are known on the host and the device.

Exercise 2:

The Arnoldi-process is a common strategy to compute parts of the eigenvalue spectra of a large (sparse-) matrix $A \in \mathbb{R}^{n \times n}$. The algorithm produces an orthonormal matrix $Q \in \mathbb{R}^{n \times k}$

$$Q = \begin{bmatrix} q_1 & q_2 & \cdots & q_k \end{bmatrix}$$

which is used to project the original matrix A to

$$\tilde{A} = Q^T A Q$$

from which the eigenvalues can be extracted using classical dense eigenvalue solvers.
The Arnoldi-process to compute Q from a start vector r_0 is given by the following algorithm:

1: **for** $k \in \{1, 2, \ldots\}$ and $r_{k-1} \neq 0$ **do**
2: $\quad h_{k,k-1} \leftarrow \|r_{k-1}\|_2$
3: $\quad q_k \leftarrow \frac{r_{k-1}}{h_{k,k-1}}$
4: $\quad r_k \leftarrow A q_k$
5: $\quad$ **for** $j = 1, \ldots, k$ **do**
6: $\quad\quad h_{jk} \leftarrow \langle q_j, r_k \rangle$
7: $\quad\quad r_k \leftarrow r_k - q_j h_{jk}$
8: $\quad$ **end for**
9: **end for**

Formulate an asynchronously working pseudo code algorithm which computes the matrix Q on the device and solves the eigenvalue problem for $\tilde{A}$ on the host. The algorithm should stop either if a deflating subspace, i.e. $r_k = 0$, was found or the ℓ, $\ell < n$, largest eigenvalues of $\tilde{A}$ have converged.