ulm university universität
uulm

Institut of Numerical Mathematics
Dr. Andreas F. Borchert and Dr. Michael C. Lehn
Mladjan Radic

27 October 2017
Quiz 1

High Performance Computing I (WS 2016/2017)

Deadline: 3 November 2017, 2pm

Notice:

Please register yourself in SLC for HPC I. The registration will allow you to submit your answers to the following questions. Please type your responses in two simple text files named "quiz01.txt" and "quiz01.c". Generate the gnuplot as described and add "quiz01.png". Submit these files to our server *thales* using the following command:

```
thales$ submit hpc quiz01 quiz01.txt quiz01.c quiz01.png
```

Question 1

In BLAS, the GEMM operation is defined as

$$C \leftarrow \beta C + \alpha AB, \qquad \alpha, \beta \in \mathbb{F},\ A \in \mathbb{F}^{m \times k},\ B \in \mathbb{F}^{k \times n},\ C \in \mathbb{F}^{m \times n}$$

where $\mathbb{F}$ denotes a set of floating point numbers (e.g. single or double precision floating point numbers).

Special cases for the operation are specified as follows:

- If $\beta = 0$ the initial values of C are ignored and the operation behaves like $C \leftarrow \alpha AB$.
- If $k = 0$ the formal matrix product AB is defined as a $m \times n$ zero matrix.

Formulate two algorithmic variants for the GEMM operation:

1. The variant *gemm_rowMajor* accesses the entries of all matrices always row-wise.
2. The variant *gemm_colMajor* accesses the entries of all matrices always column-wise.

As an example for writing an algorithm consider:

```
initMatrix
==========
Input:   A = (a_{i,j})_{0<=i<m, 0<=j<n}
Result:  A is overwritten as specified in [Paper2018]

for i = 0,..., m-1
    for j = 0,..., n-1
        a_{i,j} <- i*n + j + 1
```

Question 2

1. Implement the two algorithmic variants through functions called *dgemm_rowMajor* and *dgemm_colMajor*
2. Write a program that tests and benchmarks the two implementations for different matrix sizes. For testing it is sufficient to check whether both algorithms produce the same numerical results.

Please note:

- The implementation is supposed to match the algorithms, e.g. variable names in the implementation are consistent with the notation in the algorithm.
- The code should follow a consistent coding style. You have *some* freedom to follow your very own coding style. However, it at least should comply to the following restrictions:
  - The code has a proper indentation. And yes, it should be clear what proper means!
  - No line has more than 80 characters.

Question 3

Send us a nice graphics that displays the result of the benchmarks. Obviously, the term *nice* includes at least the following points:

- There is a title that states what this benchmark is about.
- Axes are labeled.
- There is a legend that describes what different curves represent.
- Any kind of text can be read with old and tired eyes.