Assignment 2

Relevant concepts

Stack allocation:
- From Assignment 0 we know that there are two types of allocated memory. Stack allocation is generally faster but it can't hold as large data as heap allocation. Stack allocation might be useful because of its speed compared to heap allocation.

Heap allocation:
- Might be useful when stack allocation is not large enough.

Avoiding memory fragmentation:
- It might be useful to avoid memory fragmentation as this allows to faster run through matrixes.

Writing to files:
- Writing to files uses a number of commands such as fprintf, fwrite, fopen, fclose and so on. This will be important as writing to files is an important part of the assignment.

Parsing command line arguments:
- Parsing command lines is of importance since the program ”newton” is to be started with arguments.

Measuring time:
- Time measurements will be used on individual subtasks as well as the whole program to analyze the speed and optimization of the program and subtasks.

Inlining:
- Inlining is that a subtask is written in the main code instead of in a sub-program. This generally results in a more slipstreamed compiled code which results in a faster program. This should be implemented when possible.

Locality and Indirect addressing:
- In assignment 1 we learned that the runtime is much smaller when accessing the memory in the right order. This is basically the same for indirect addressing, we want to access the slots in an array close to the currently used element while it is still in easily accessible memory.

Valgrind/GDB
- Valgrind and/or GDB will be used in the optimization process and bug solving.

Intended program layout

The program can be divided in four subtasks. Parsing arguments, thread management, Newton's method computation and writing to files. The initial idea is to use one thread (if available) to write to the files and the rest to do the computations.

1. Parsing command line arguments for the variables that will be used throughout the program.
2. Splitting up to different threads and syncing.
3. Making sure that data is transfered between the different threads in an efficient way.
4. Checking the convergence and divergence conditions
5. Computing Newton's method
6. Writing to files.

Where point 1 compares to the parsing of arguments, 2 and 3 is the thread management, 4 and 5 is the newton's method and 6 is writing to files.