Direct Solvers for FEM systems

For the d-dimensional model problem with an analogous node numbering as in Example 1.43,

$$K_h \in \mathbb{R}^{n_h \times n_h}, \qquad n_h = \mathcal{O}(h^{-d}), \qquad \text{band width } \mathcal{O}(h^{-(d-1)}).$$

Gauss (LU factorization) exploiting band structure:

	$d=1$	$d=2$	$d=3$
operations	$\mathcal{O}(n_h)$	$\mathcal{O}(n_h^2)$	$\mathcal{O}(n_h^{7/3})$
memory for storing L, U	$\mathcal{O}(n_h)$	$\mathcal{O}(n_h^{3/2})$	$\mathcal{O}(n_h^{5/3})$

The huge memory consumption stems from the fact that although K_h is sparse, the factors L, U are not. This phenomenon is called *fill-in*.

Question: Are there better numberings that reduce the memory complexity?

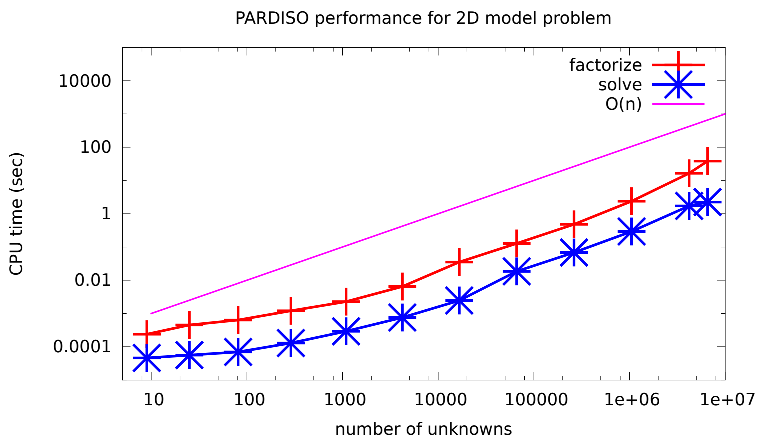
Answer: Yes. The optimal reordering can even be computed in optimal complexity. However, the factorization of the reordered system is still not optimal:

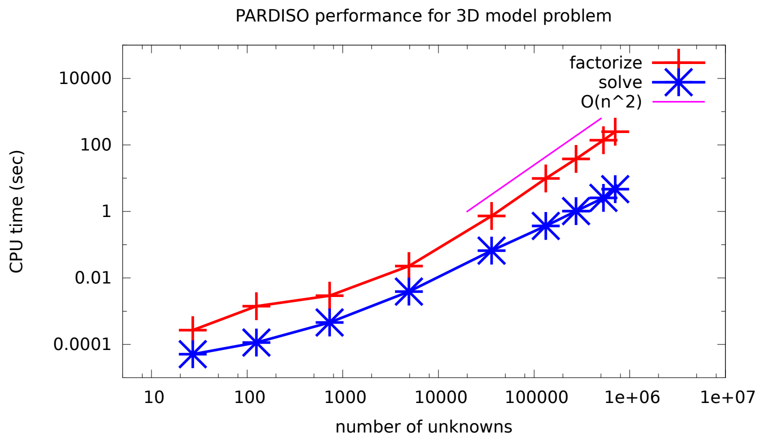
Optimal reordering and Gauss exploiting band structure:

	$d=1$	$d=2$	$d=3$
operations	$\mathcal{O}(n_h)$	$\mathcal{O}(n_h^{3/2})$	$\mathcal{O}(n_h^2)$

For Poisson's equation and structured grids (like in Example 1.43), the 2D complexity can even be shown to be $\mathcal{O}(n_h \log^\alpha(n_h))$ (quasi-optimal). In 3D, the quadratic behavior is sharp.

Performance Study. The following graphs show the performance of the solver package PARDISO (embedded in the Intel Math Kernel library) for the 2D/3D model problem.





The computations were carried out on a notebook with an Intel Core i5-520M processor (2.40 GHz) and 8 GByte RAM.

The curves 'factorize' correspond to the CPU times needed for computing the factors L and U, the curves 'solve' correspond to the CPU time needed for solving the two triangular systems with L and U. The plots use a double-logarithmic scale in order to show the almost linear behavior in 2D and the quadratic one in 3D.

We see that the solution of 3D problems is soon problematic. The problem with 100 nodes in each direction ($n_h \approx 10^6$) could not be solved (within the given RAM of 8 GByte).