「計算機実験 I」実習課題 (EX3)

- 講義のページ: `http://exa.phys.s.u-tokyo.ac.jp/ja/lectures/2017S-computer1`
- サンプルプログラム:
  `https://github.com/todo-group/computer-experiments/tree/master/exercise/linear_system`
- 準備練習
  1. ベクトルや行列を扱うためのユーティリティー関数が (`matrix_util.h`) に用意されている。サンプルプログラム `matrix_example.c` の中身を見て、その使い方を確認せよ。
  2. LU 分解のサンプルプログラム (`lu_decomp.c`) をコンパイル・実行せよ。コンパイル時に LAPACK をリンク (`-llapack`) する必要があることに注意 (ハンドブック 3.1.6 節)。

     ```
     $ cc lu_decomp.c -o lu_decomp -llapack
     $ ./lu_decomp input1.dat
     ```

- 基本課題
  1. `lu_decomp.c` を参考にして、LU 分解を用いて行列の行列式を計算するプログラムを作成せよ。$n \times n$ の Vandermonde 行列 ($v_{ij} = x_j^{i-1}$) ($x_1 \cdots x_n$ は実数) の行列式を計算し、厳密な値 $\prod_{1 \le i < j \le n} (x_j - x_i)$ と比較せよ。
  2. LU 分解を用いて Dirichlet 型の境界条件のもとでの二次元 Laplace 方程式の解を求めるプログラムを作成せよ。適当な境界条件 [例えば $u(0,y) = \sin(\pi y)$, $u(x,0) = u(x,1) = u(1,y) = 0$] を仮定して解を計算し、Gnuplot の `splot` コマンドを用いて解をプロットせよ。また、メッシュ数を増やすと、解の形や計算時間がどのように変化するか調べよ (計算時間の測り方については、ハンドブック 2.1.6 節参照)。
  3. Laplace 方程式の境界値問題を Jacobi 法で解くプログラムを作成せよ。メッシュ数を増やしていった場合の計算速度を隣の学生のプログラムと比較し、速度差の原因を考察せよ。
- 応用課題
  1. C 言語におけるポインタの振る舞いをテストするプログラム (`pointer.c`) のソースコードを見て、どのような出力が生成されるか予想せよ。実際にコンパイル・実行して予想を確かめてみよ。
  2. Laplace 方程式の境界値問題を Gauss-Seidel 法、SOR 法で解くプログラムを作成し、計算結果や計算速度を LU 分解・Jacobi 法と比較せよ。また、収束までの回数を Jacobi 法と比較せよ。特に SOR 法の場合、パラメータ ω の選び方により、どのように収束回数が変化するか観察し、最適な ω の値について考察せよ。
  3. photon では、OS 付属の LAPACK 以外にも、Intel 製の MKL (Math Kernel Library) に含まれる LAPACK ルーチンが利用可能である[1]。ガウスの消去法のプログラム (`gauss.c`)、標準 LAPACK を使った LU 分解 (`lu_decomp.c`)、MKL を使った LU 分解の速度を比較せよ (特に $n = 1000$ 以上の大きな行列について)。なぜ、速度に大きな差が生じるのか、MKL で使われている (と予想される) 最適化手法について調べてみよ。

[1] MKL を使うには、GNU C コンパイラ (`cc`, `gcc`) の代わりに Intel C コンパイラ (`icc`) を使い、`-llapack` の代わりに `-mkl` を指定してリンクを行えばよい。例: `icc lu_decomp.c -o lu_decomp -mkl`