「計算機実験 I」実習課題 (EX3)

- サンプルプログラム: example-1-L3.zip
- 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 (`cmatrix.h`) に用意されている。サンプルプログラム `matrix_example.c` や行列・行列積を計算するプログラム `multiply.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 と BLAS をリンク (`-llapack -lblas`) する必要がある (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。(ピボット選択で行を入れ替えると、行列式の符号が反転することに注意)
  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 $u(0,y) = \sin(2\pi y)$, $u(1,y) = \sin(\pi y)$, $u(x,0) = u(x,1) = 0$] や電荷分布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. 非線形連立方程式

     $$f(x,y) = x^2 + y^2 - 1 = 0$$
     $$g(x,y) = x^2(2+x) - y^2(2-x) = 0$$

     を多次元のニュートン法 (L1 スライド p.20) を用いて解け。ヤコビ行列の逆行列をかける代わりに、LU 分解で線形連立方程式を解くこと

- 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-matrix.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. 行列・行列積の計算を行うサンプルプログラム `multiply.c` と BLAS ライブラリを使ってそれと等価な計算を行う `multiply_dgemm.c` の速度を比較せよ[1]。行列サイズによっては数十倍もの性能差が出ことがあるが、BLAS で使われている最適化手法について調べてみよ

[1] ai では、OS 付属の BLAS、LAPACK ではなく、Intel 製の MKL (Math Kernel Library) に含まれる BLAS や LAPACK を利用するのがよい。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 -lblas` の代わりに`-mkl` を指定してリンクする。例: `icc -O3 multiply_dgemm.c -mkl`