実対称固有値問題に対する多分割の分割統治法の分散メモリ型並列計算機への実装

田村純一 坪谷怜 桑島豊 重原孝臣

埼玉大学大学院理工学研究科

H21 後期若手利用者推薦 課題報告会 (2010/05/21)

実対称固有値問題

- 大規模な数値シミュレーションの需要が大きい
  - 分子軌道計算 (計算化学) や構造解析 (建築) など
  - 実対称行列の固有値問題に帰着
- 大規模な実対称固有値問題を高速に解きたい
  - 対象の行列を三重対角化してから解くのが一般的

実対称三重対角固有値問題の数値解法

- 二分法・逆反復法 (BII)
  $O(n^2)$，高並列性[1)]
- 二分割の分割統治法 [Cuppen 80] (DC2)
  $O(n^3)$，高精度，主要演算は行列積のため並列性高
- 多分割の分割統治法 [桑島ら 06] (DCK，分割数 $k \geq 2$)
  DC2 の拡張，行列積の演算量が $n^3 \longrightarrow (2/k)n^3$
  - 逐次実装，共有並列化 [田村ら 09] は提案済
  - 分散並列化 (H21 前期に開始)

目的：T2K 環境で分散並列 DCK の実装・実験を行う

[1)] n：入力行列サイズ

分散並列 DCK の概要

入力： n 次実対称三重対角行列 T，分割数 k
出力：直交行列 Q，対角行列 Λ ($T = Q\Lambda Q^T$ を満たす)

1. $T = P(D + UU^T)P^T$ と同値変形．D は対角，U は $n \times (k-1)$，P は k 個の直交行列の直和
2. 固有分解 $D + UU^T \equiv \tilde{Q}\Lambda\tilde{Q}^T$ を計算
   - 2.1 $D + UU^T$ の自明な $n - r$ 個の固有対を除去
   - 2.2 固有値を各プロセスが r/p 個ずつ分散並列計算[2)]
   - 2.3 固有ベクトルを各プロセスが r/p 本ずつ分散並列計算
   - 2.4 非直交な固有ベクトル同士をまとめてグループ化 (閾値 ϵ)
   - 2.5 グループ毎に再直交化
3. 行列積 $Q \equiv P\tilde{Q}$ を分散並列計算

[2)] p をプロセス数とし，各プロセスを p_j と呼ぶ．

前回[3)]からの変更点

- 固有ベクトル計算の負荷を各プロセスに均等に分散
  - 旧：先頭のプロセスから n/p ずつ担当し，
    末尾のプロセスは端数を取る
  - 新：全プロセスが r/p ずつ担当

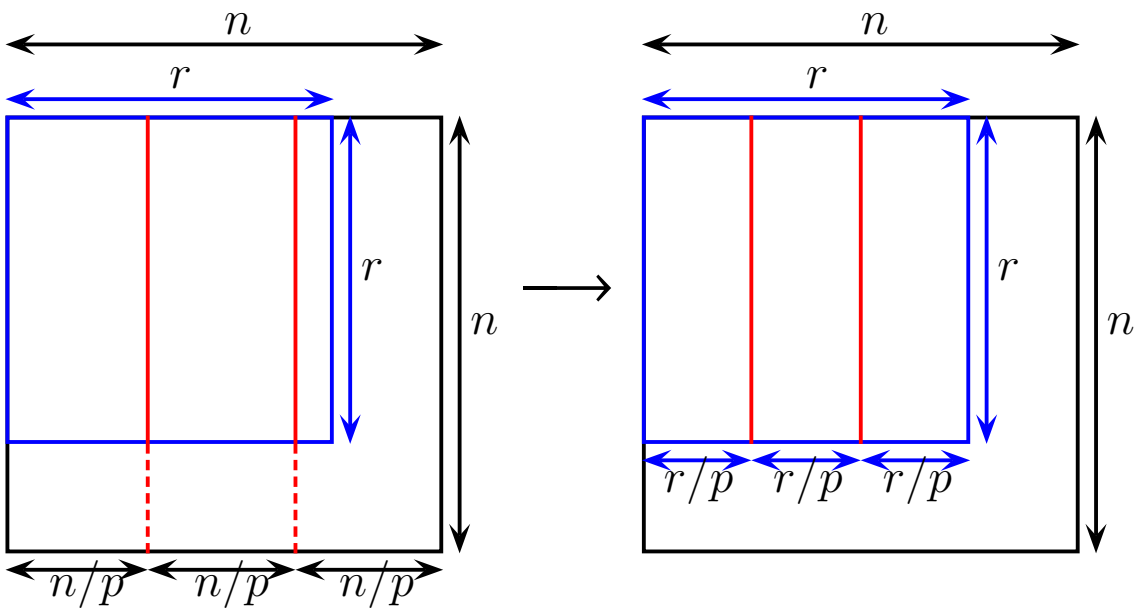


図 1: 固有ベクトル計算の割り振り方の改良

[3)] H21 前期

前回からの変更点

- 小グループの再直交化を他のプロセスも担当
  - 旧：小グループの再直交化は p_0 のみ担当
  - 新：他のプロセスも担当

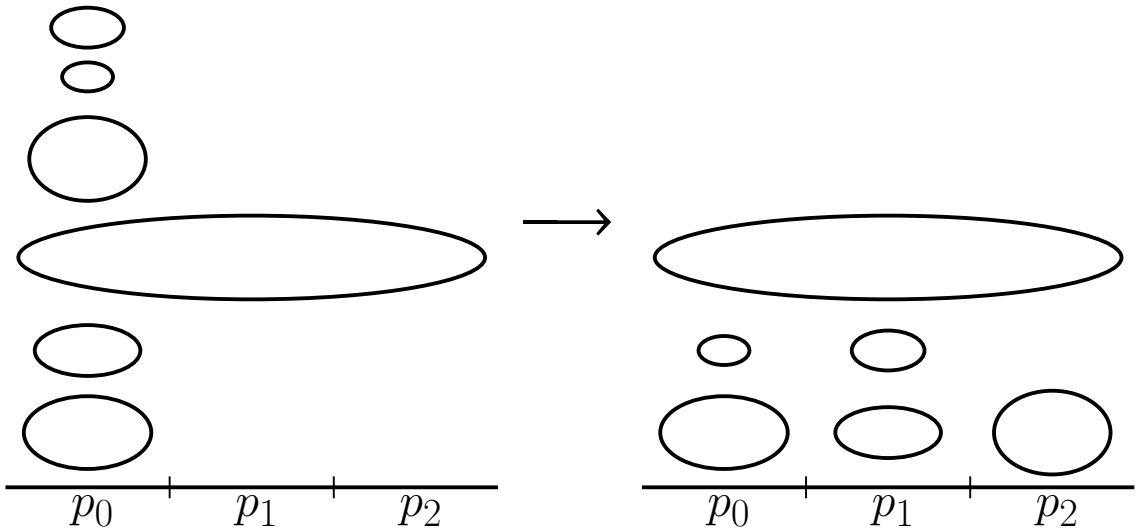


図 2: 再直交化の負荷分散の改良[4)]

[4)] 楕円がグループを表し，大きさがグループ内のベクトル数を示す

前回からの変更点

- 行列積を PBLAS により分散並列計算
  - 旧：行列データの通信と積の計算を MPI+BLAS で実装
  - ブロックサイズを大きく取る場合は PBLAS を用いた方が有効，という簡易ベンチマークの結果により，PBLAS を採用

数値実験

- 入力行列 T の数値的固有対 $(\lambda_j, \boldsymbol{q}_j)$ の計算時間と精度を，提案手法と標準的な数値線形代数ライブラリ ScaLAPACK に実装された解法 (BII，DC2) と比較
- 精度を次式で評価：

$$\text{相対残差 } \epsilon_r = \max_{1 \le j \le n} \frac{\|T\boldsymbol{q}_j - \lambda_j \boldsymbol{q}_j\|_2}{\|T\|_2}$$

$$\text{直交誤差 } \epsilon_o = \max_{1 \le i \le j \le n} |\boldsymbol{q}_i^T \boldsymbol{q}_j - \delta_{ij}|$$

- 入力行列：

  QC: 量子カオス系をモデル化した行列

  SQ: 行列 QC を対角に二つ並べて繋ぎ目に 1 を入れた行列

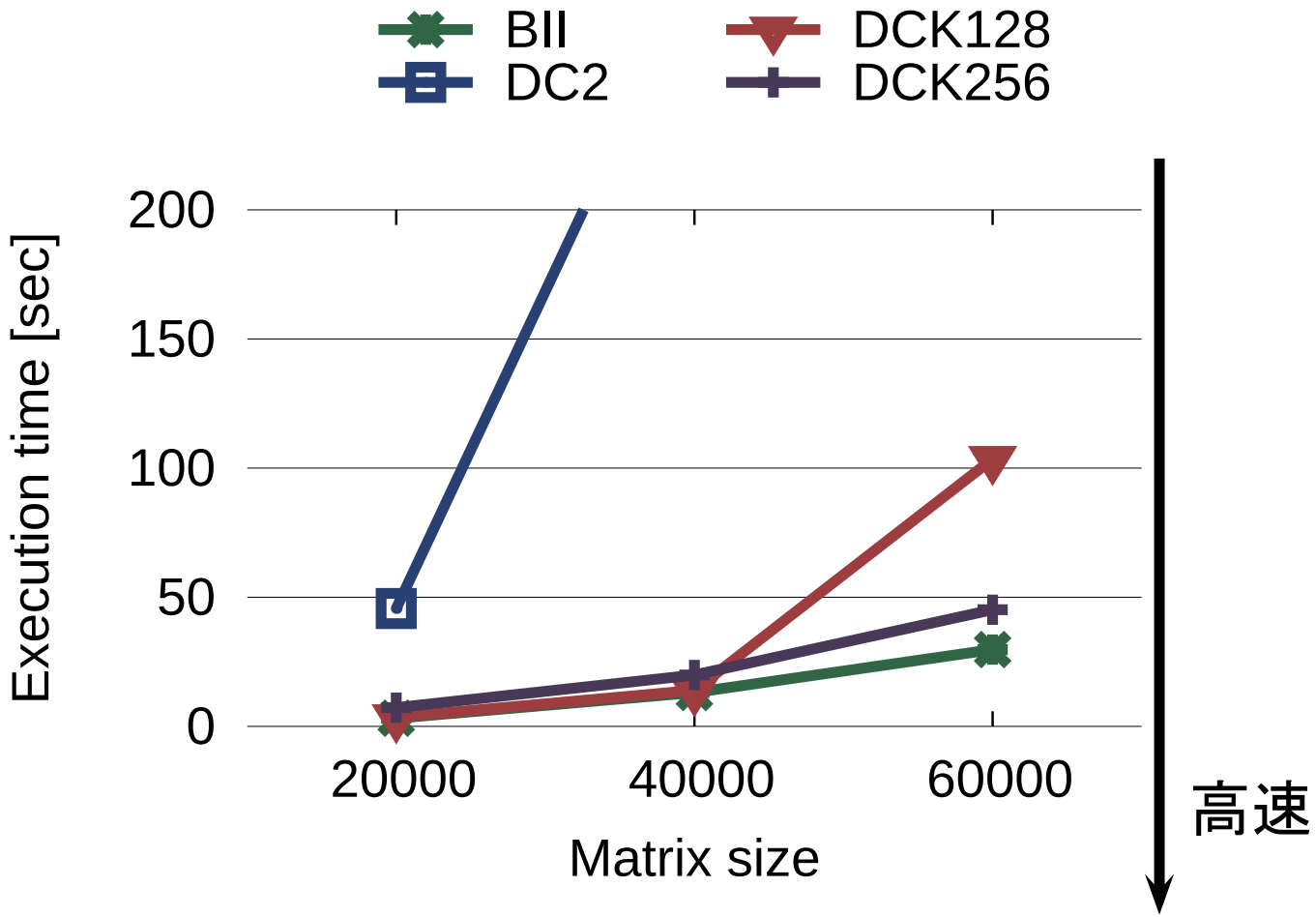
  DS: 対角成分が副対角成分と比べて小さい行列

実験環境

- 分散メモリ型並列計算機 HITACHI HA8000 を
  8 ノード (128 CPU) 使用 (東京大学情報基盤センター)
- 128 プロセス使用し，　DCK の分割数は $k = 128, 256$ を指定
- DCK の基礎的な線形代数演算に，
  並列計算機に備え付けの数値線形代数ライブラリ
  (ScaLAPACK/PBLAS, LAPACK/BLAS) を利用
- DCK の再直交化の閾値 ϵ を緩めた[5] ($\epsilon =$ 5e-13)

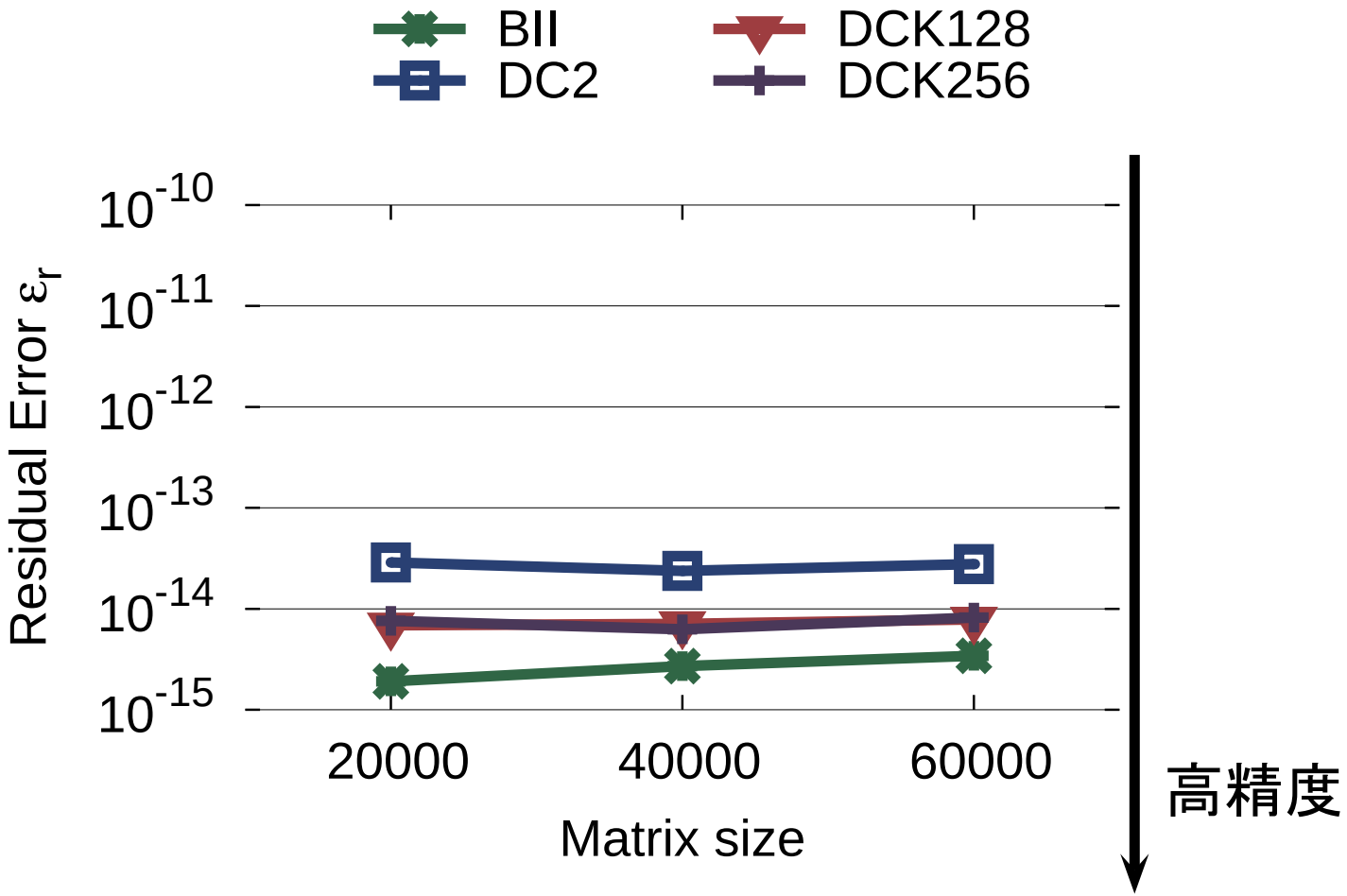
[5] 前回は $\epsilon =$ 5e-14

実験結果：計算時間 (行列 QC)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,
DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

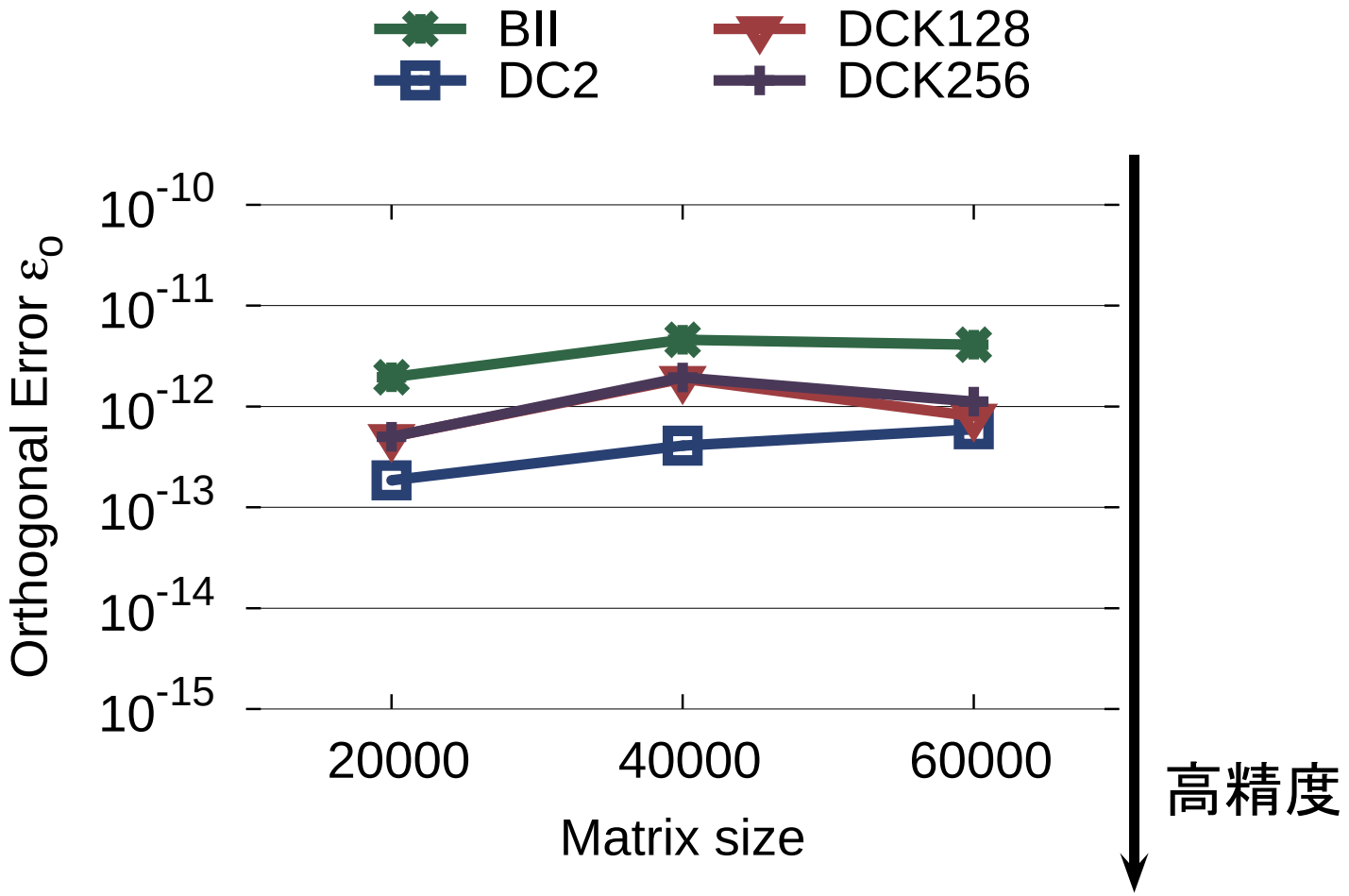
実験結果：相対残差 (行列 QC)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,

DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

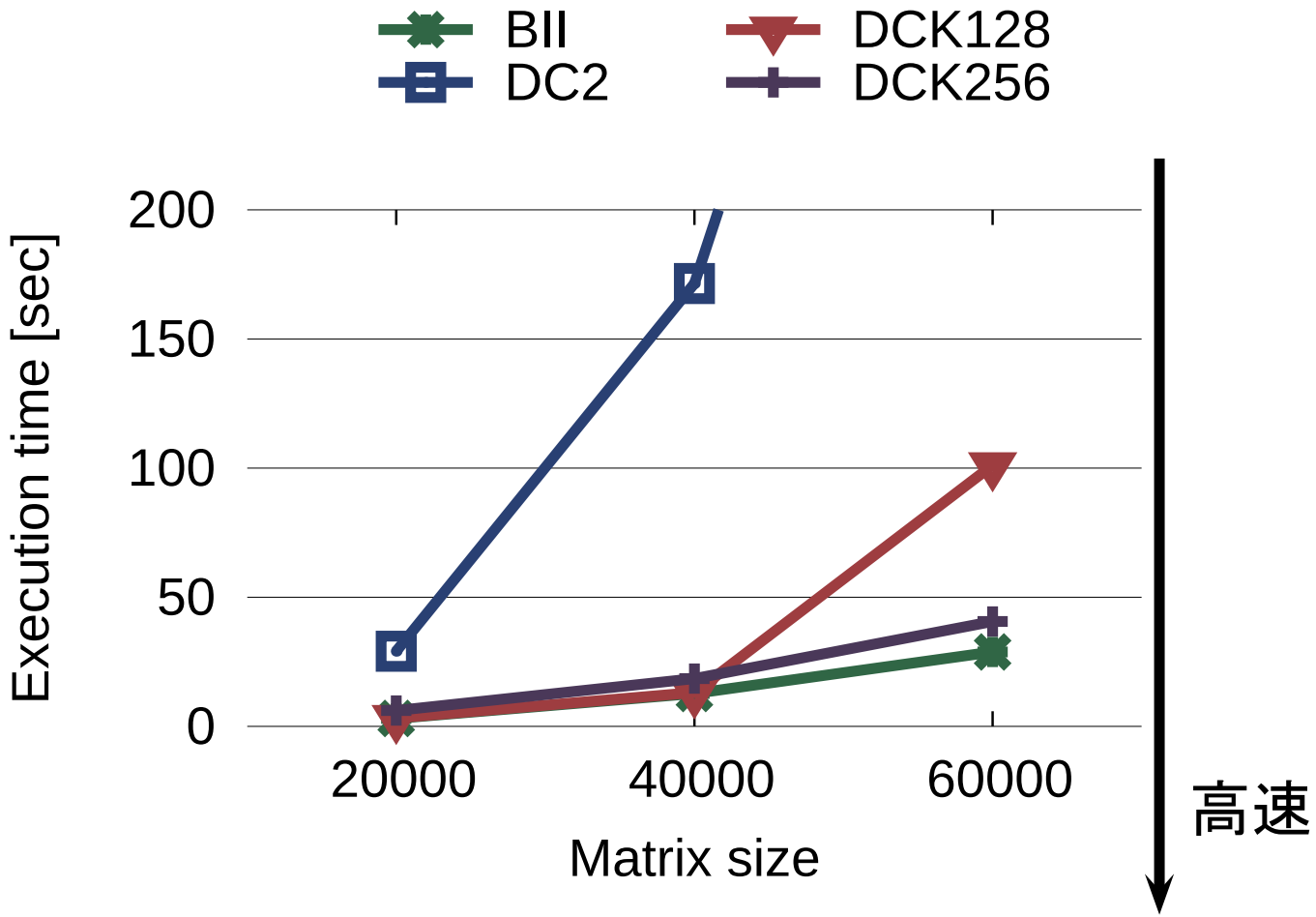
実験結果：直交誤差 (行列 QC)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,

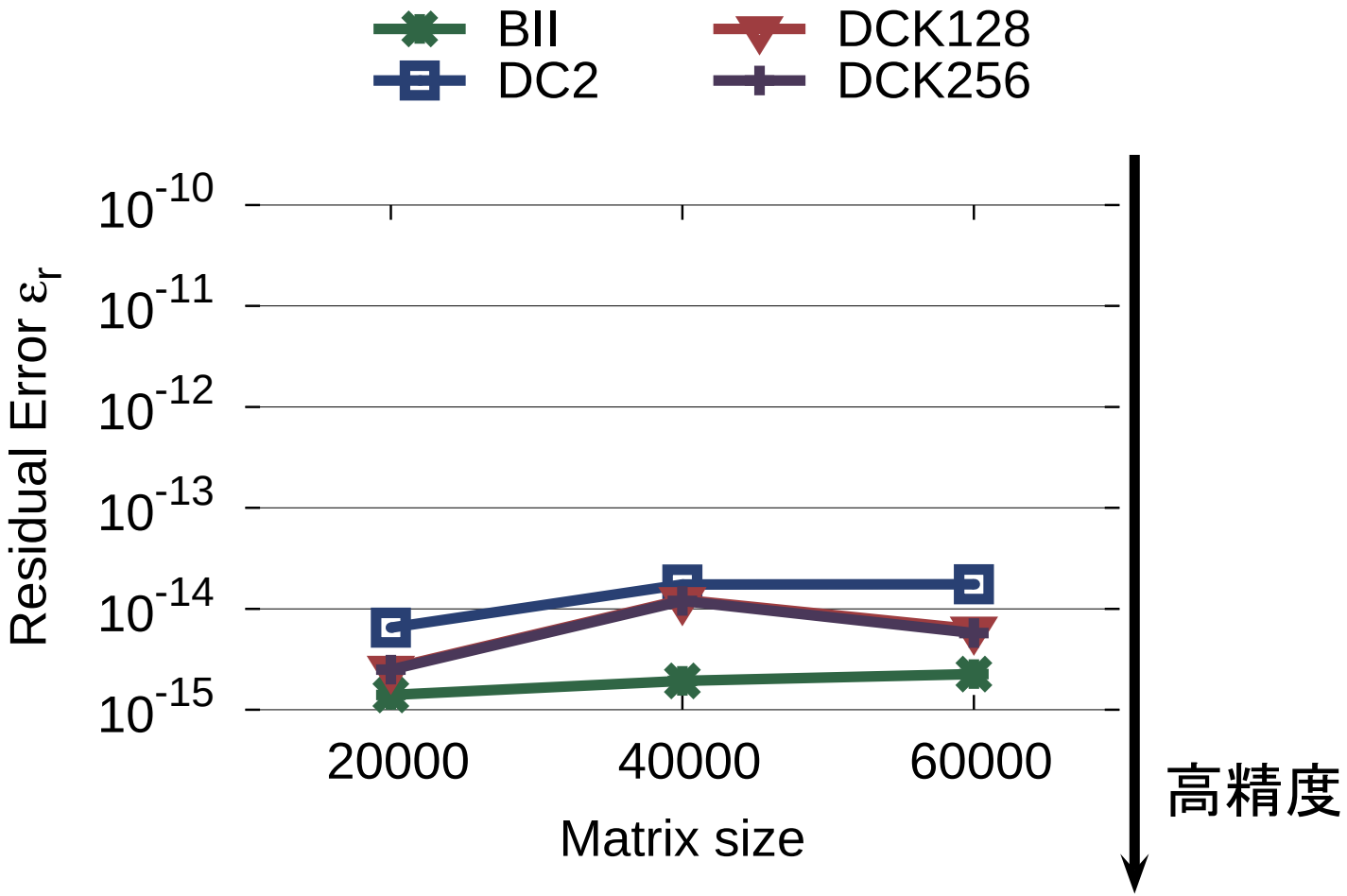
DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

実験結果：計算時間 (行列 SQ)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,
DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

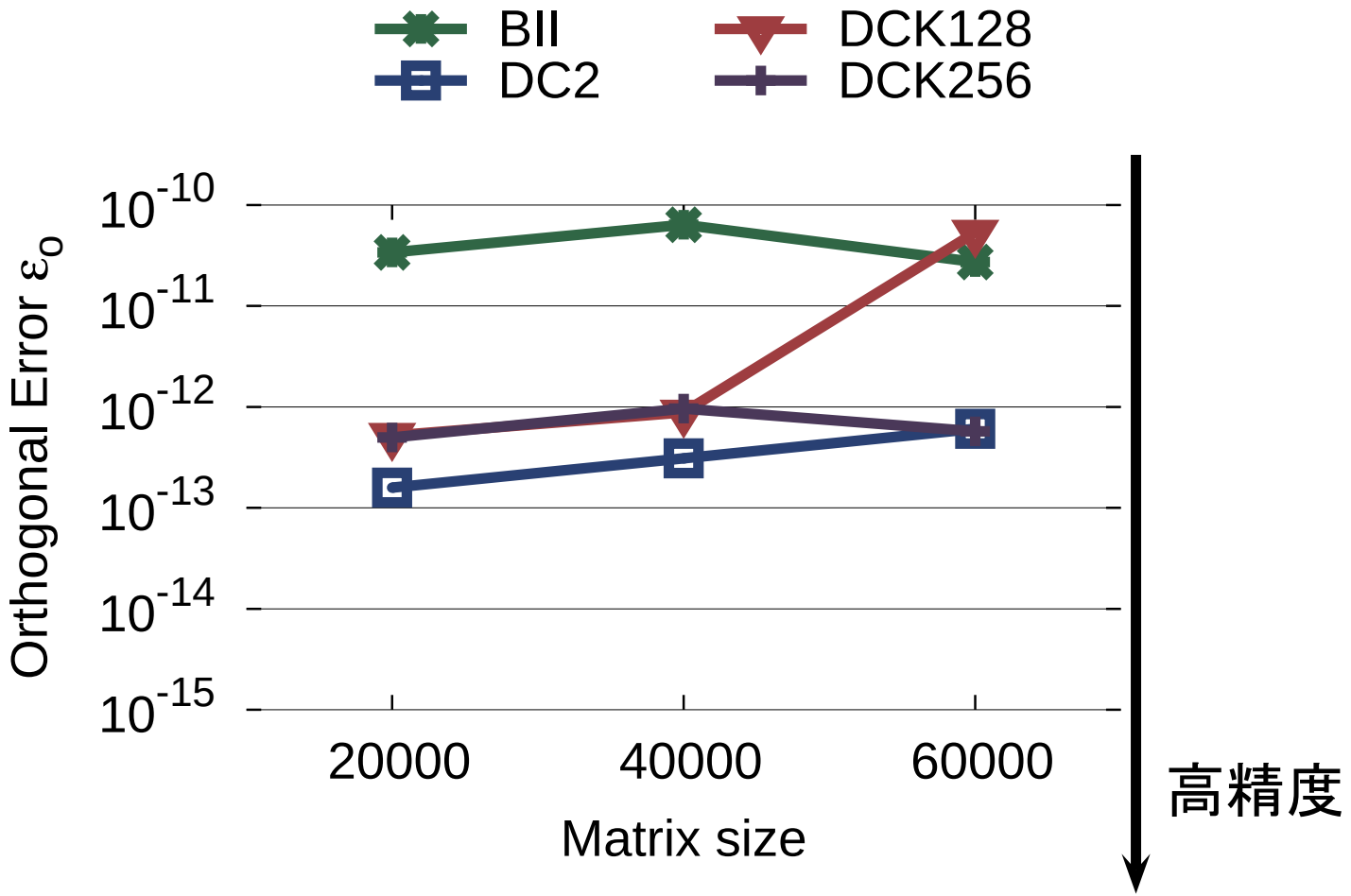
実験結果：相対残差 (行列 SQ)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,

DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

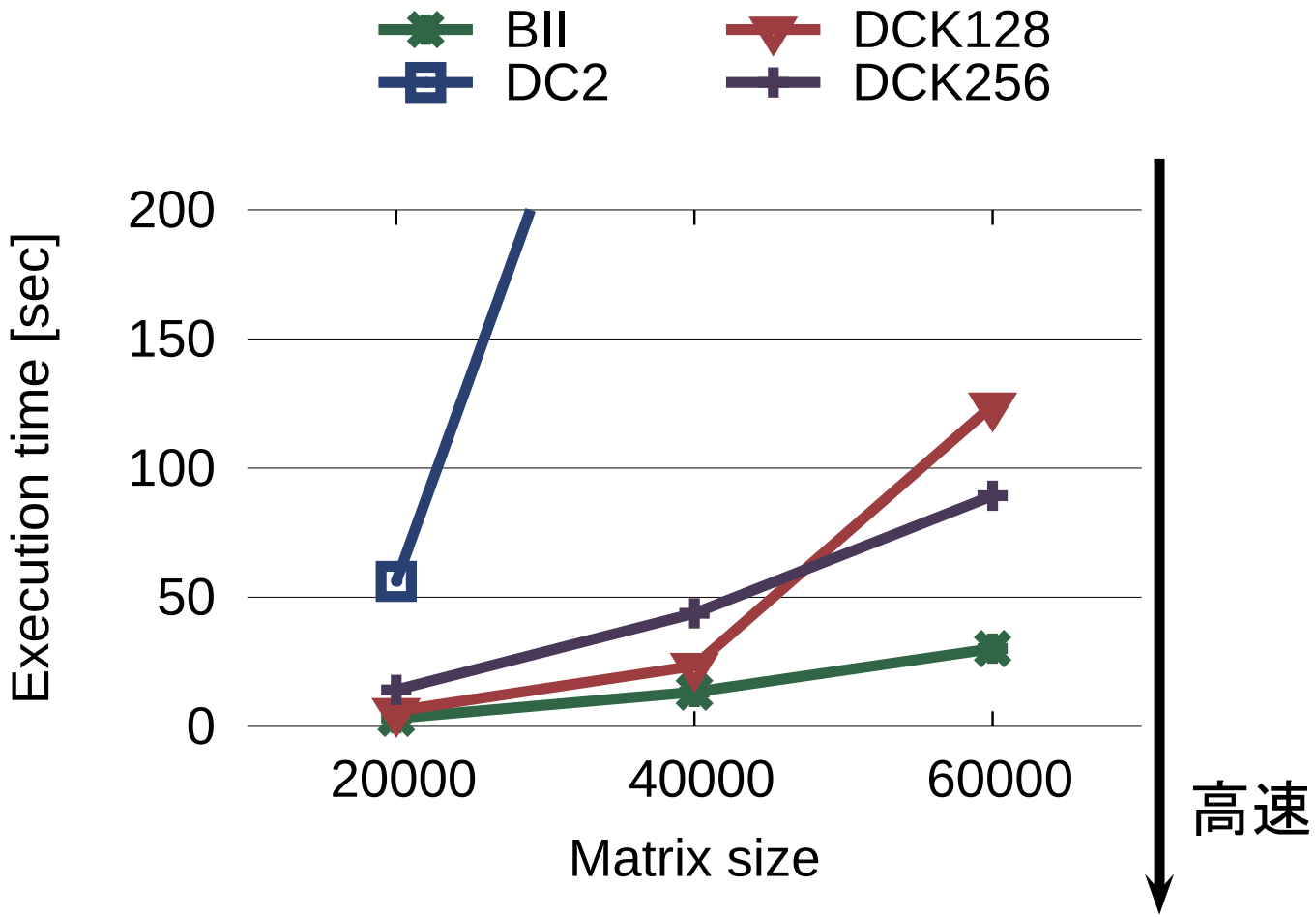
実験結果：直交誤差 (行列 SQ)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,

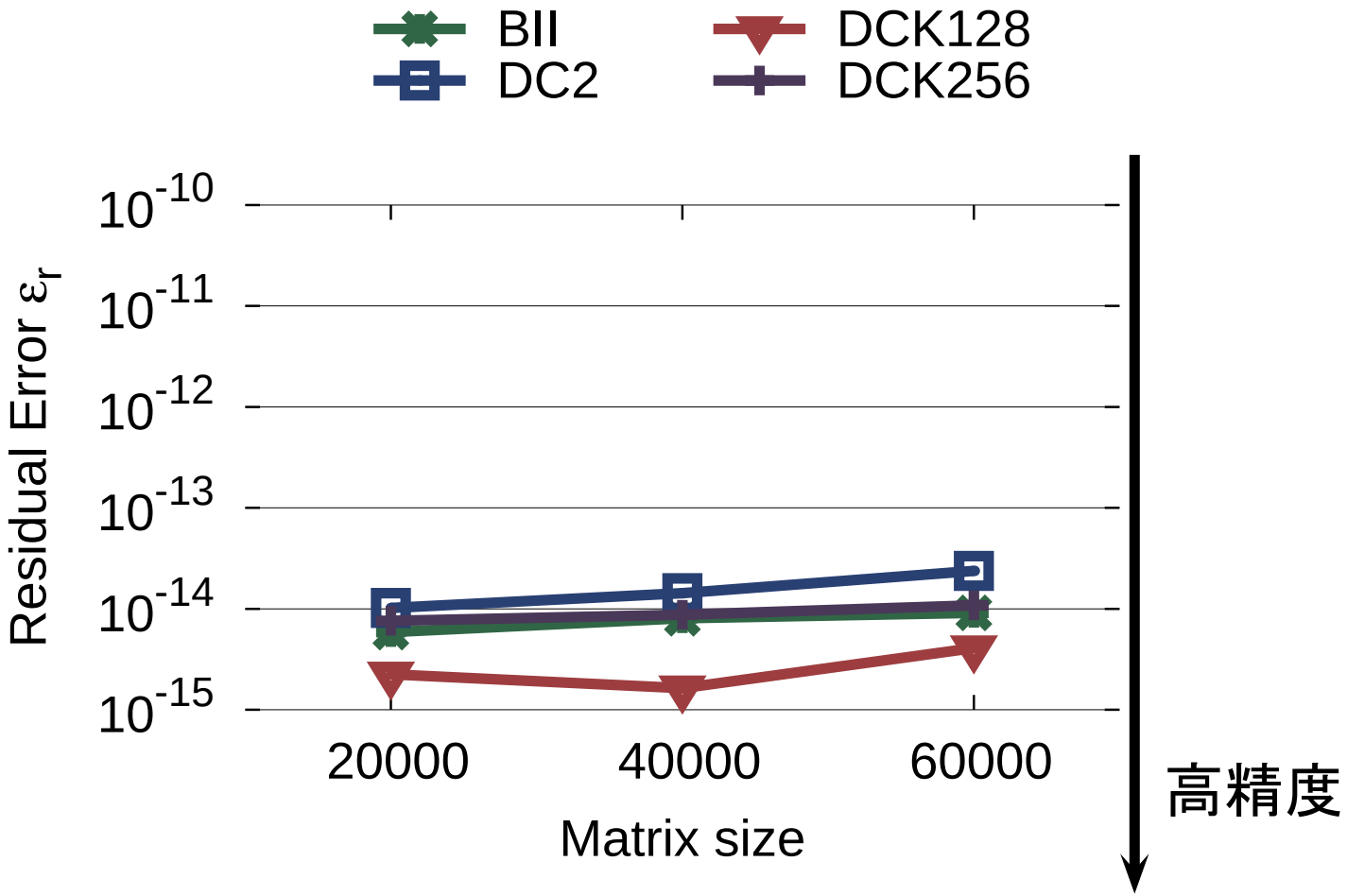
DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

実験結果：計算時間 (行列 DS)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,
DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

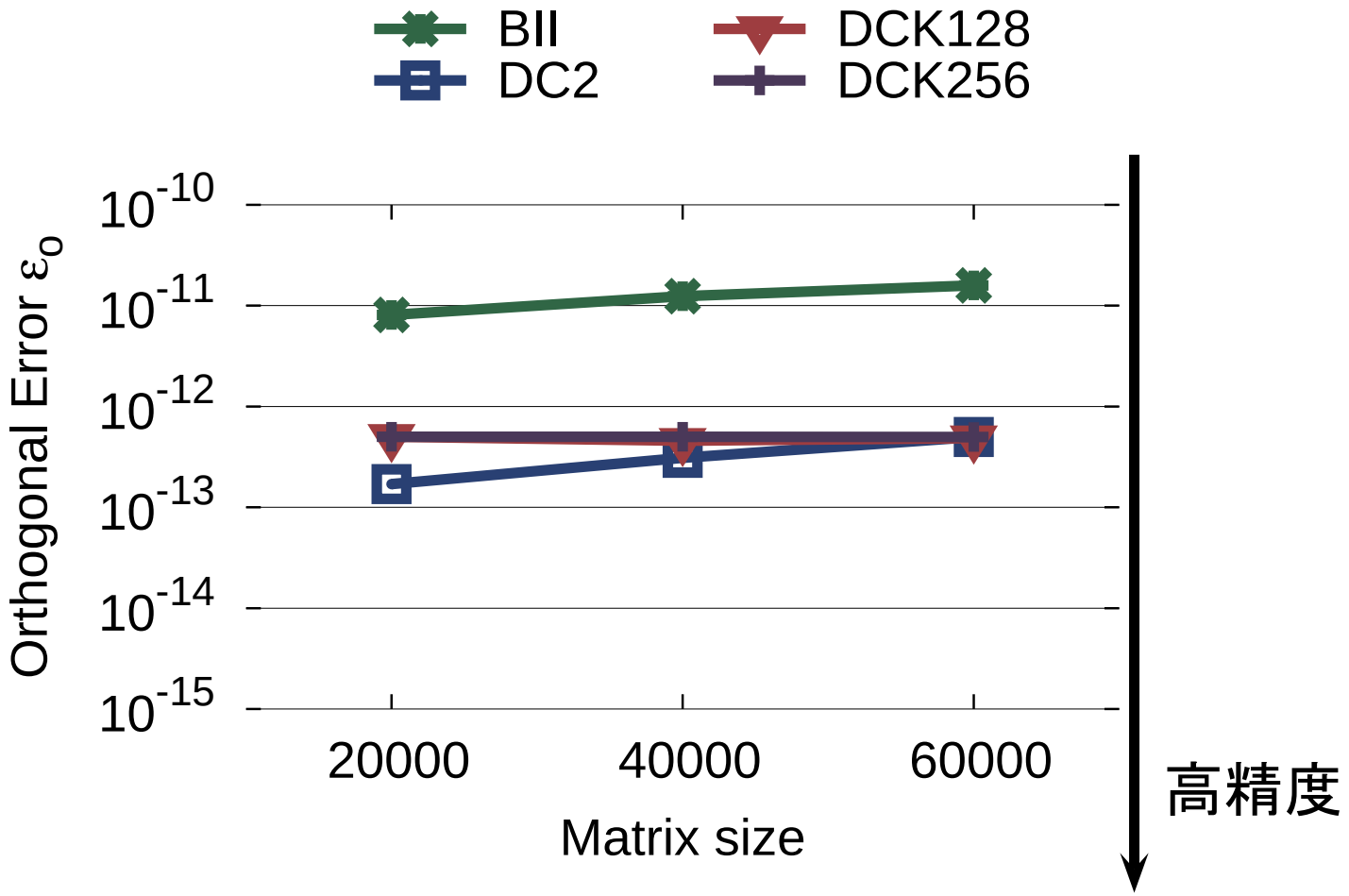
実験結果：相対残差 (行列 DS)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,

DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

実験結果：直交誤差 (行列 DS)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,

DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

まとめ

- 多分割の分割統治法 (DCK) の分散並列実装を T2K 環境に移行
  - 固有ベクトル計算や再直交化の負荷分散を改良
  - 8 ノード (128CPU) 利用し，行列サイズを 60000 次まで拡大
- 計算時間
  - 二分法・逆反復法 (BII) と同程度だが，二分割の分割統治法 (DC2) の $1/5$ 以下の時間
  - 再直交化の閾値 ϵ に依存．適切な閾値の設定方法が必要
- 計算精度
  - 直交誤差は，行列 SQ/DS に対しては BII より $1 \sim 2$ 桁良く，行列 QC に対しては同等か若干良い．いずれも DC2 とは同程度．
  - 相対残差はいずれの解法も十分小さい
- 精度を重視する場合に DCK が有効

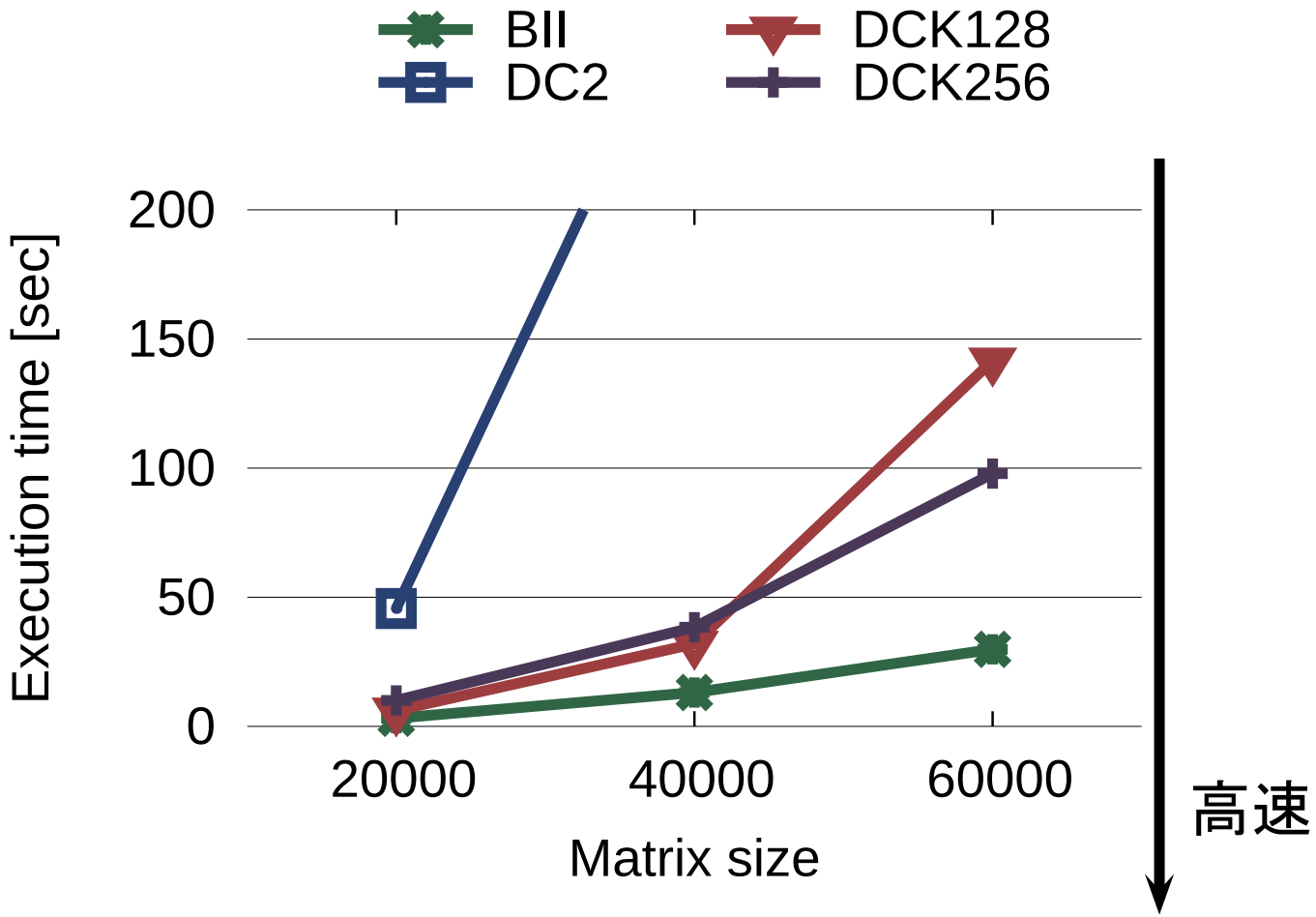
謝辞：
東京大学情報基盤センターの若手利用者推薦制度による支援に感謝申し上げます．

成果

- 田村純一，坪谷怜，桑島豊，重原孝臣: 実対称固有値問題に対する多分割の分割統治法の分散並列アルゴリズムの提案, 情報処理学会論文誌コンピューティングシステム (ACS), (採択済, 未発行).

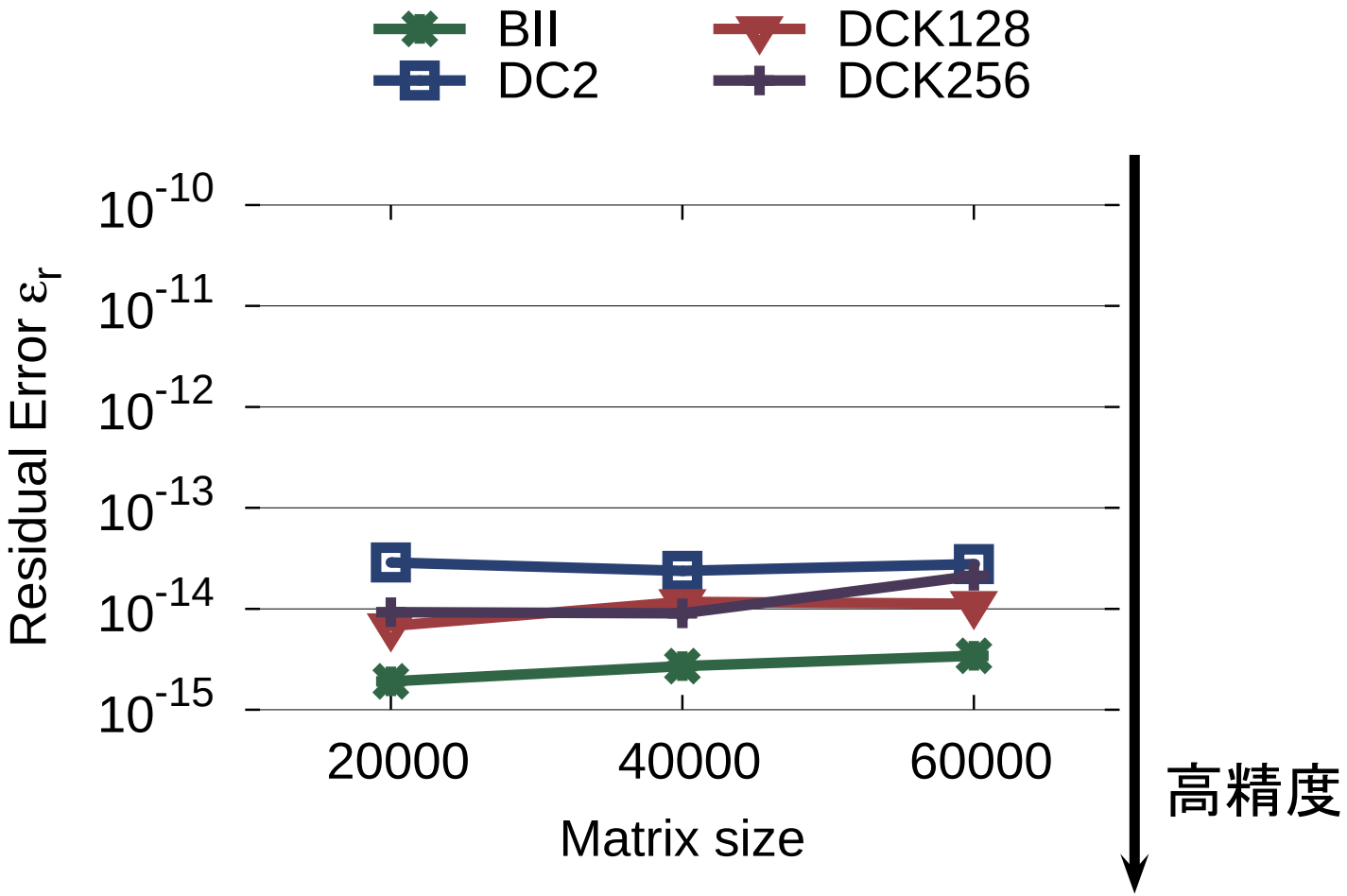
Appendix

実験結果：計算時間 (行列 QC)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,
DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

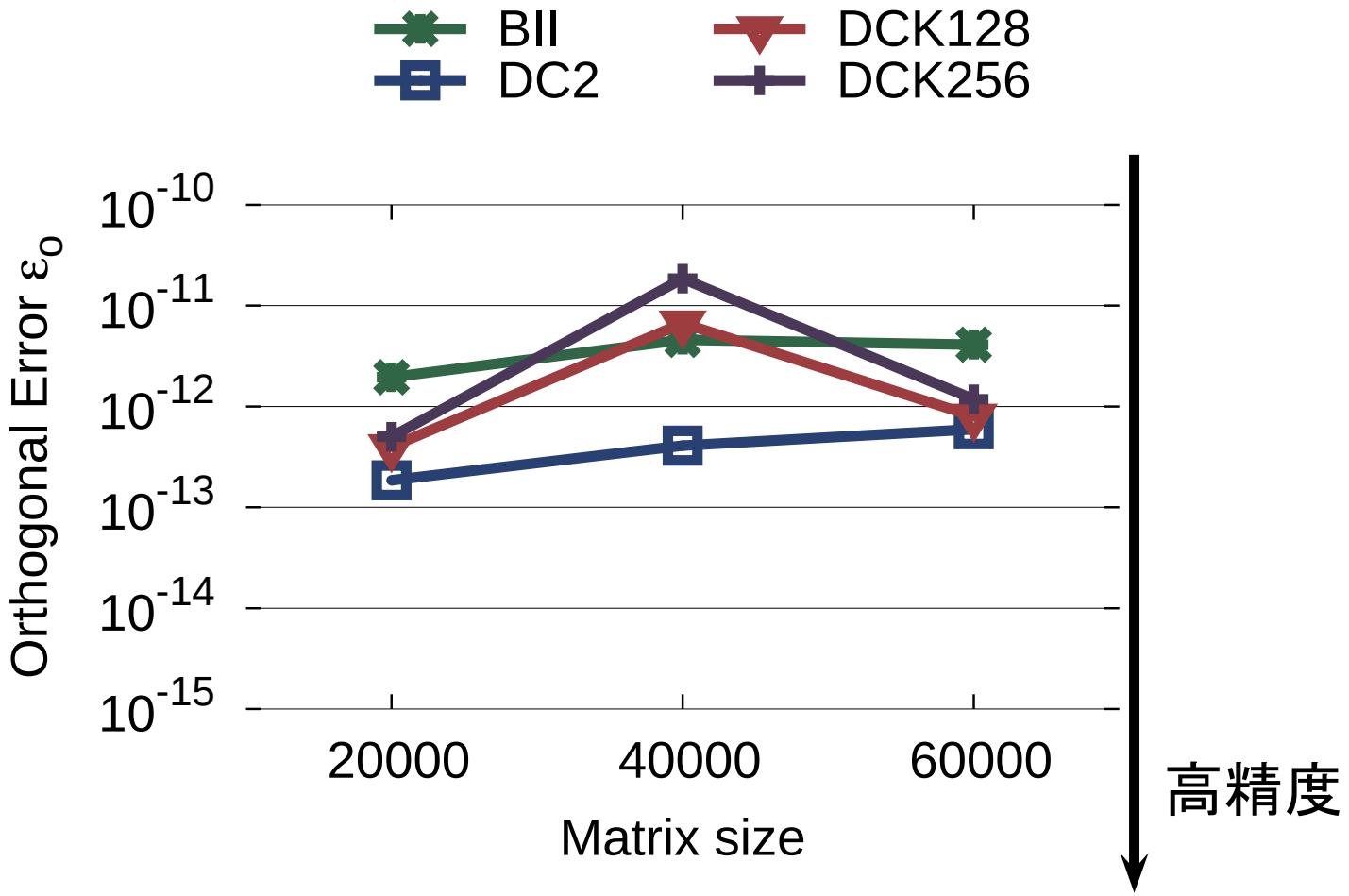
実験結果：相対残差 (行列 QC)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,

DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

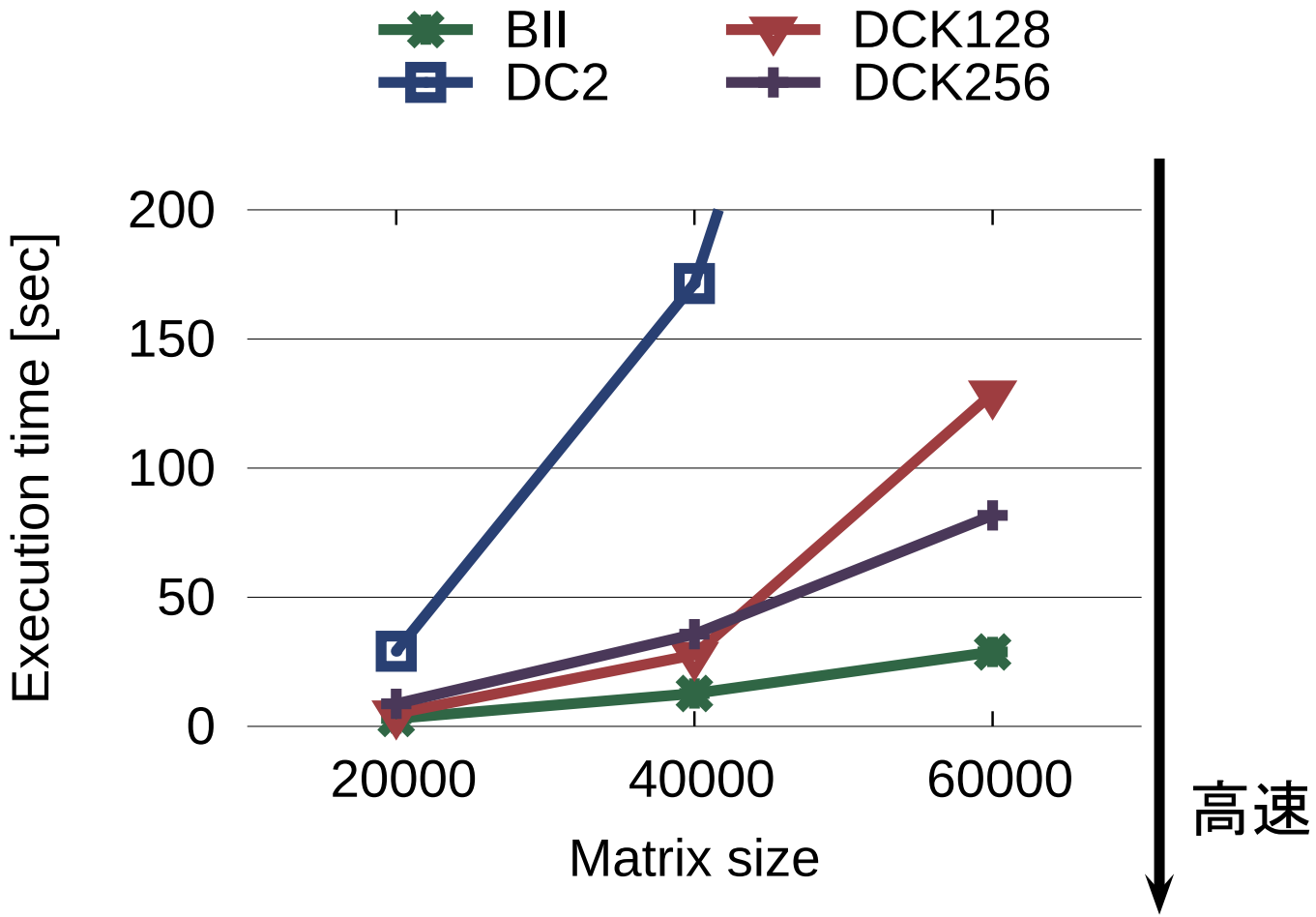
実験結果：直交誤差 (行列 QC)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,

DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

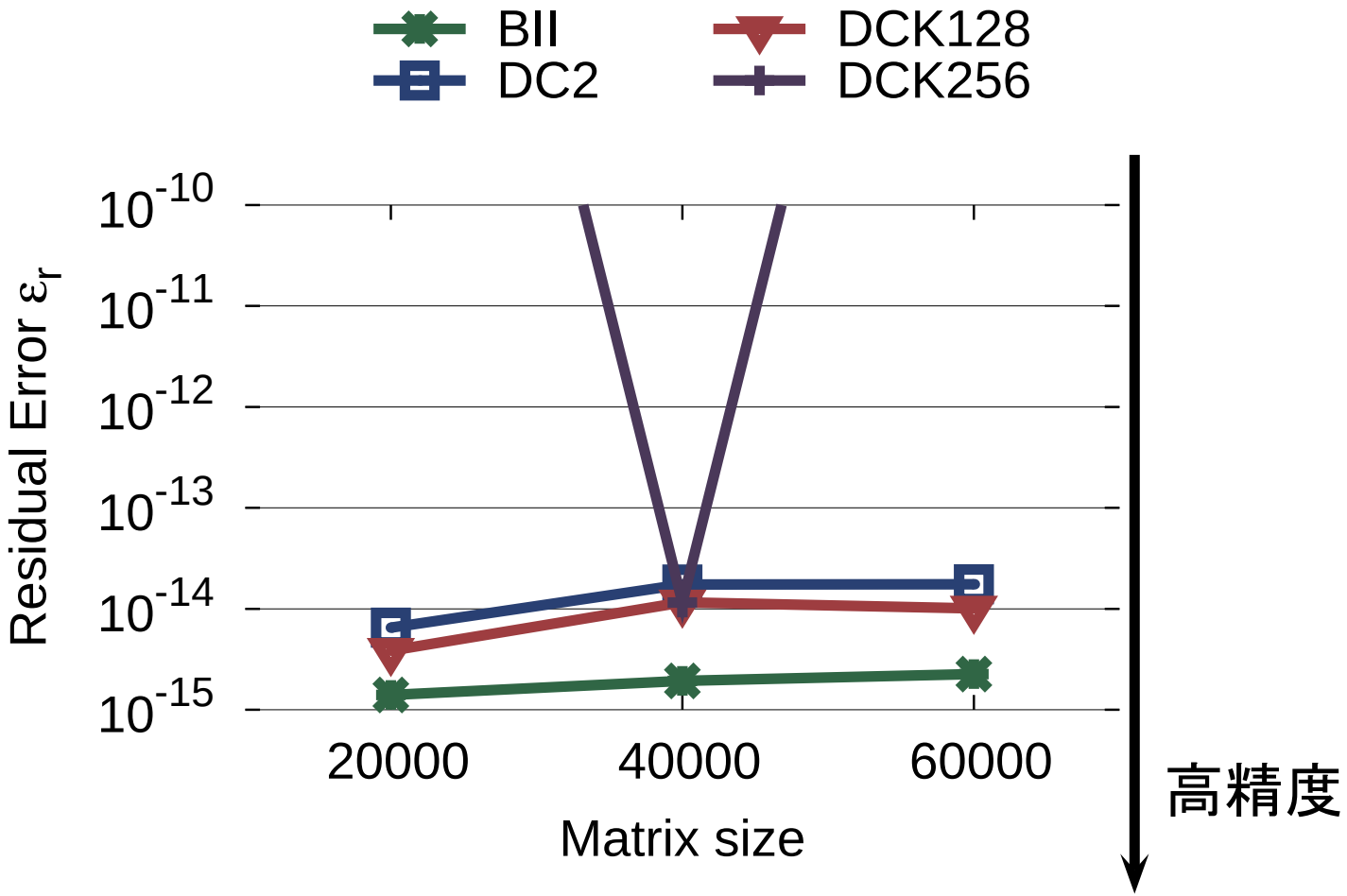
実験結果：計算時間 (行列 SQ)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,

DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

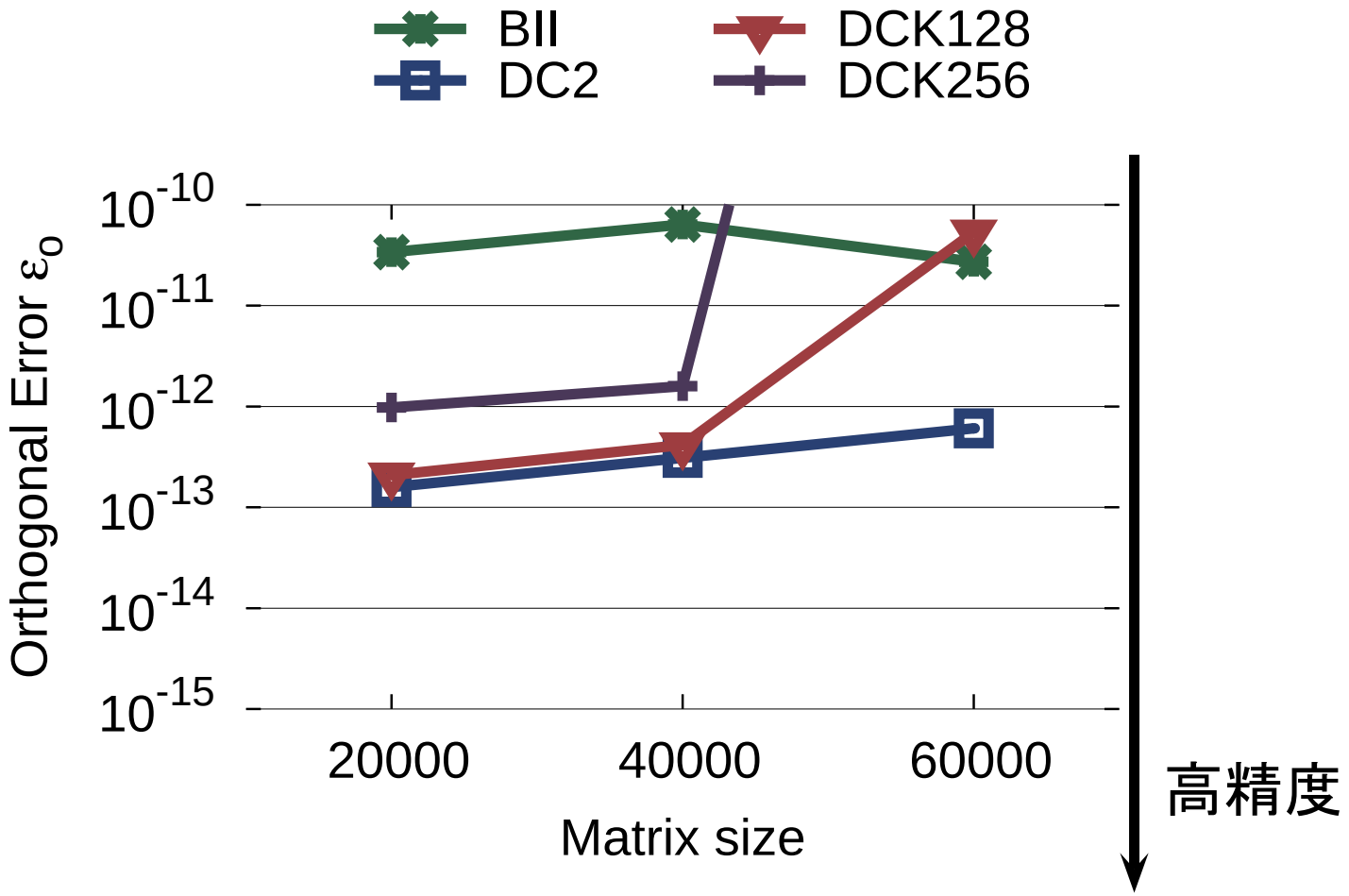
実験結果：相対残差 (行列 SQ)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,

DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)

実験結果：直交誤差 (行列 SQ)



BII: 二分法・逆反復法, DC2: 二分割の分割統治法,

DCK128: 提案手法 ($k = 128$), DCK256: 提案手法 ($k = 256$)