応用数理工学特論 2006/06/15

計算理工学専攻 学籍番号 280667094 高橋 翔馬
計算理工学専攻 学籍番号 280667167 兵藤 一永

2006 年 6 月 27 日

1 固有値計算の概要

- 固有値計算とその応用
- 固有値・固有ベクトル計算の流れ
- 各部分の演算量
- 三重対角行列 T に対する固有値・固有ベクトル解法

標準固有値問題 $Ax = \lambda x (A$: 実対象 $(a_{ij} = a_{ji})$ またはエルミート行列) の固有値と固有ベクトルを求めるとする．これらは，分子計算，統計計算，構造解析などに用いられる．
固有値問題の区分として以下のようなものがある．

- 対称 or 非対称
- 密行列 or 疎行列
- 固有ベクトルが必要か?(必要な場合は全部が必要か?それとも一部が必要か?)

今回は対称かつ密行列のみを扱う

1.1 固有値・固有ベクトル計算の流れ

1. 実対称行列 A を三重対角化する (ハウスホルダー法)　　$[Q^*AQ = T]$
2. 三重対角化行列の固有値・固有ベクトルを計算 (QR 法，DC 法，二分法・逆反復法，Dhillon のアルゴリズム) これで，T の固有値 λ_i と固有ベクトル y_i を求める　　$[Ty_i = \lambda_i y_i]$
3. 三重対角化行列の逆変換を行う　　$[x_i = Qy_i]$
4. A の固有ベクトル x_i を求める

実対称行列 A には n^2 個の要素があるので，三重対角化行列にして固有値計算の計算量を減らし，ここから固有ベクトルを導出する．

1.2 各部分の演算量

- N × N の実対称行列 A に対し，M 組の固有値・固有ベクトルを求める場合

!

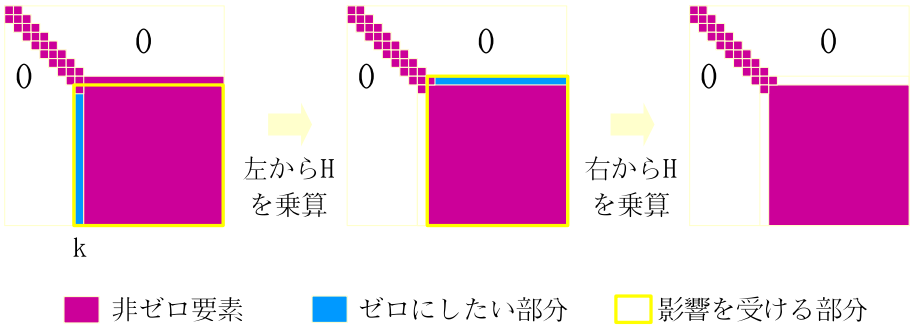

図 1　第 k ステップでの処理

  - ハウスホルダー方による三重対角化:　$(4/3)N^3$
  - 三重対角行列の固有値・固有ベクトルの計算:　$O(NM)$〜$O(N^3)$
    (解法および固有値の分布により大きく異なる)
  - 逆変換:　$2N^2M$
- $M << N$ の場合，三重対角化の演算量が支配的であり，この部分の並列化，最適化が重要

2　三重対角化処理の並列化

2.1　ハウスホルダー法による三重対角化

- 鏡像変換による列の消去
  - 鏡像変換 $H = I - \alpha uu^t$
  - H は対称な直行行列
  - 与えられたベクトルの第 1 成分以外をゼロにする
- 第 k ステップでの処理を図 1 に示す

2.2　ハウスホルダー法の特徴

- 演算量は N^3 のオーダであり，また演算はほとんどが BLAS2 のため，行列データの再利用性がない．
  - ブロック化など，アルゴリズム上の工夫が必要

2.3　共有メモリ向けの並列化

行列ベクトル積および rank-2 更新の部分について，BLAS2 レベルで並列化する (並列 BLAS を使えば，並列化は極めて容易)．BLAS2 はデータ再利用性がないためバスを通した共有メモリへのアクセスが頻発する．アクセス競合を避けるためには，キャッシュ向け最適化技法が有効である．

- ブロックサイクリック列分割による並列化

!

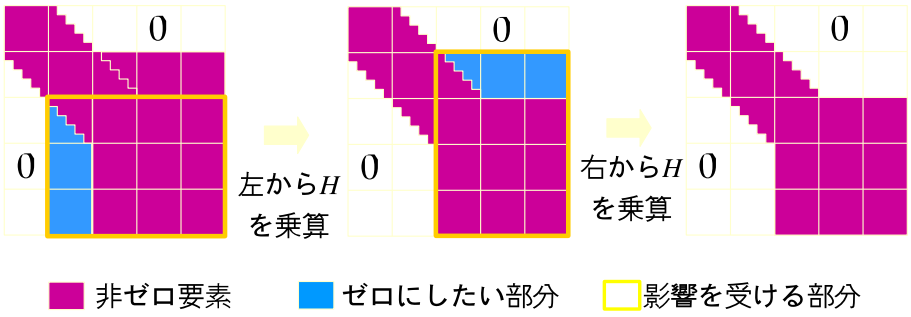

図 2　第 k ステップでの処理

  - 行列を幅 b の列ブロックに分割し，ブロックを周期的にプロセッサに割り当てる
  - 通信時間は $O(Nlogp)$
- Scattered Square 分割による並列化
  - 行列を b*b のブロックに分割し，ブロックに対して大きさ $\sqrt{p} * \sqrt{p}$ のプロセッサテンプレートを周期的に割り当てる
  - 通信時間は $O(Nlogp/\sqrt{(}p))$
- 各 PU の計算時間はともに $O(N^2/p)$
- PU が増えるほどブロック SS 分割が有利

3　三重対角化処理のキャッシュ向け最適化

従来のハウスホルダー法では行列データの再利用性がないため，キャッシュマシンでは高い性能が期待できず，ブロック化などアルゴリズムの工夫が必要.

■Bischof のアルゴリズム　蜜行列 A を半帯幅 L の帯行列 B に変換し，この帯行列を三重対角行列 T に変換する.

- 帯行列への変換は BLAS3 を利用可能
- 帯行列から三重対角行列への変換の演算量は $O(N^2L)$ で前半部よりずっと小さい

1. K=1 から $N/L-1$ まで以下の 2.〜6. を繰り返す
2. $D^{(k)}$ を第 1 ブロックが上三角行列で第 2 ブロック以下がゼロ行列であるブロックベクトルに変換するブロック鏡像変換 $I - U^{(k)}\,\alpha^{(k)}U^{(K)t}$ を求める
3. 行列・ブロックベクトル積：$P^{(K)} = C^{(k)}U^{(k)}\,\alpha^{(K)}$
4. ブロックベクトルの内積：$\beta^{(K)} = \alpha^{(k)}U^{(k)t}P^{(K)}/2$
5. ブロックベクトルの加算：$Q^{(K)} = P^{(k)} - U^{(k)}\,\beta^{(K)}$
6. 行列の rank-2L 更新：$C^{(K)} = C^{(k)} - U^{(k)}Q^{(K)t} - Q^{(k)}U^{(K)t}$

■Bischof のアルゴリズムの特徴と問題点

  - データの再利用性

    * 演算のうち $O(N^3)$ の部分はすべて BLAS3 化
    * キャッシュサイズに応じて L を選ぶことで，データ再利用性を最大化することが可能
  - 計算精度
    * 多彩な例題を用いて体系的に評価した結果はまだ報告されていない
  - 演算量
    * 後半部の演算量は L に比例して増加
    * 全体としての実行時間が最小になるように L を適切に選ぶ必要がある

4 性能・精度評価

- キャッシュマシン向け三重対角化アルゴリズムとしては W_u のアルゴリズムが最高速であり，大規模問題では Dongarra のアルゴリズムの約 2 倍の性能が達成できる．
- 特に $N = 3840$ の場合，W_u のアルゴリズムでは Opteron, Alpha21264A でピークの 50 %以上の性能が達成できる．
- Bischof/W_u のアルゴリズムによる固有値の相対誤差は，Dongarra のアルゴリズムに比べてやや大きい．しかし増大の程度は多くの場合 2〜3 倍以内，最大でも 1 桁程度である．

5 今後の展開

- より多様な例題での精度評価
- 固有ベクトル計算部分の実装と性能・精度評価
- 共有／分散メモリ型計算機上での並列化と性能評価
- 性能パラメータ L, L' の自動最適化