

(19) 日本国特許庁(JP)

再公表特許(A1)

(11) 国際公開番号

W02008/018188

発行日 平成21年12月24日 (2009.12.24)

(43) 国際公開日 平成20年2月14日 (2008.2.14)

(51) Int.Cl. F I テーマコード (参考)  
**G06F 17/16 (2006.01)** G06F 17/16 K 5B056

審査請求 未請求 予備審査請求 未請求 (全 67 頁)

出願番号	特願2008-528725 (P2008-528725)	(71) 出願人	504132272 国立大学法人京都大学
(21) 国際出願番号	PCT/JP2007/051575		京都府京都市左京区吉田本町36番地1
(22) 国際出願日	平成19年1月31日 (2007.1.31)	(74) 代理人	100115749 弁理士 谷川 英和
(31) 優先権主張番号	特願2006-215660 (P2006-215660)	(72) 発明者	中村 佳正 京都府京都市左京区吉田本町 国立大学法人京都大学大学院情報学研究科内
(32) 優先日	平成18年8月8日 (2006.8.8)	(72) 発明者	坪井 洋明 京都府京都市左京区吉田本町 国立大学法人京都大学大学院情報学研究科内
(33) 優先権主張国	日本国 (JP)	(72) 発明者	譽田 太朗 京都府京都市左京区吉田本町 国立大学法人京都大学大学院情報学研究科内

最終頁に続く

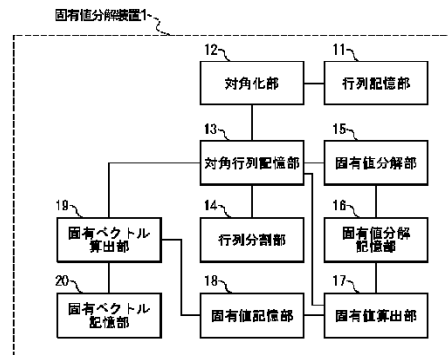
(54) 【発明の名称】 固有値分解装置、及び固有値分解方法

(57) 【要約】

【課題】高速・高精度であり、並列処理可能な固有値分解装置を提供する。

【解決手段】対称3重対角行列Tを2個の対称3重対角行列に分割する処理を繰り返し行う行列分割部14と、分割後の対称3重対角行列を固有値分解する固有値分解部15と、固有値分解部15によって固有値分解された固有値と、固有ベクトルからなる直交行列の一部の要素である行列要素とから、分割元の対称3重対角行列の固有値と、分割元の対称3重対角行列の行列要素とを算出する処理を、対称3重対角行列Tの固有値を算出するまで繰り返す固有値算出部17と、対称3重対角行列Tとその固有値とから、ツイスト分解法を用いて対称3重対角行列Tの固有ベクトルを算出する固有ベクトル算出部19と、を備える。

【選択図】 図1



- 1...EIGENVALUE DECOMPOSING DEVICE
- 12...DIAGONALIZING SECTION
- 11...MATRIX STORAGE SECTION
- 13...DIAGONAL MATRIX STORAGE SECTION
- 15...EIGENVALUE DECOMPOSING SECTION
- 19...EIGENVECTOR CALCULATING SECTION
- 14...MATRIX DIVIDING SECTION
- 16...EIGENVALUE DECOMPOSITION STORAGE SECTION
- 20...EIGENVECTOR STORAGE SECTION
- 18...EIGENVALUE STORAGE SECTION
- 17...EIGENVALUE CALCULATING SECTION

## 【特許請求の範囲】

## 【請求項 1】

対称 3 重対角行列 T が記憶される対角行列記憶部と、  
前記対角行列記憶部から前記対称 3 重対角行列 T を読み出し、当該対称 3 重対角行列 T を 2 個の対称 3 重対角行列に分割して前記対角行列記憶部に蓄積し、その対称 3 重対角行列を 2 個の対称 3 重対角行列に分割して前記対角行列記憶部に蓄積する処理を、分割後の各対称 3 重対角行列があらかじめ決められた大きさ以下となるまで繰り返す行列分割部と、  
前記あらかじめ決められた大きさ以下の各対称 3 重対角行列の固有値と、当該各対称 3 重対角行列の固有ベクトルからなる直交行列の一部の要素である行列要素とが記憶される固有値分解記憶部と、

10

前記あらかじめ決められた大きさ以下の各対称 3 重対角行列を前記対角行列記憶部から読み出し、前記各対称 3 重対角行列に対して固有値分解を行い、前記各対称 3 重対角行列の固有値と、前記各対称 3 重対角行列の固有ベクトルからなる直交行列の行列要素とを少なくとも算出し、当該固有値と当該行列要素とを前記固有値分解記憶部に蓄積する固有値分解部と、

前記対称 3 重対角行列 T の固有値が記憶される固有値記憶部と、  
各対称 3 重対角行列の固有値と、行列要素とを前記固有値分解記憶部から読み出し、前記固有値と、前記行列要素とから、分割元の対称 3 重対角行列の固有値と、分割元の対称 3 重対角行列の行列要素とを算出して前記固有値分解記憶部に蓄積し、その分割元の対称 3 重対角行列の固有値と、行列要素とを算出する処理を、対称 3 重対角行列 T の少なくとも 1 個の固有値を算出するまで繰り返し、前記対称 3 重対角行列 T の少なくとも 1 個の固有値を前記固有値記憶部に蓄積する固有値算出部と、

20

前記対称 3 重対角行列 T の固有ベクトルが記憶される固有ベクトル記憶部と、  
前記対角行列記憶部から前記対称 3 重対角行列 T を読み出し、前記固有値記憶部から前記対称 3 重対角行列 T の固有値を読み出し、前記対称 3 重対角行列 T とその固有値とから、ツイスト分解法を用いて前記対称 3 重対角行列 T の少なくとも 1 個の固有ベクトルを算出して前記固有ベクトル記憶部に蓄積する固有ベクトル算出部と、を備えた固有値分解装置。

## 【請求項 2】

前記固有ベクトル算出部は、q d 型ツイスト分解法により固有ベクトルを算出する、請求項 1 記載の固有値分解装置。

30

## 【請求項 3】

前記固有ベクトル算出部は、  
前記固有値記憶部から前記対称 3 重対角行列 T の固有値を読み出し、当該固有値のいずれかが負の値である場合に、前記対角行列記憶部から前記対称 3 重対角行列 T を読み出し、当該対称 3 重対角行列 T の固有ベクトルを変化させることなく、当該対称 3 重対角行列 T のすべての固有値が正の値となるように、当該対称 3 重対角行列 T を正定値化し、正定値化した後の固有値を前記固有値記憶部に蓄積し、正定値化した後の対称 3 重対角行列 T を前記対角行列記憶部に蓄積する正定値化部と、

前記対角行列記憶部からすべての固有値が正の値である対称 3 重対角行列 T を読み出し、前記固有値記憶部から前記対称 3 重対角行列 T の正の値の固有値を読み出し、前記対称 3 重対角行列 T の各要素に関して q d 型変換を行うことによって、前記対称 3 重対角行列 T を上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  にコレスキー分解するコレスキー分解部と、

40

前記上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  の各要素と、前記対称 3 重対角行列 T の正の値の固有値とを用いて固有ベクトルを算出して前記固有ベクトル記憶部に蓄積するベクトル算出部と、を備えた、請求項 2 記載の固有値分解装置。

## 【請求項 4】

前記固有ベクトル算出部は、L V 型ツイスト分解法により固有ベクトルを算出する、請求項 1 記載の固有値分解装置。

50

## 【請求項 5】

前記固有ベクトル算出部は、

前記対角行列記憶部から前記対称 3 重対角行列 T を読み出し、前記固有値記憶部から前記対称 3 重対角行列 T の固有値を読み出し、前記対称 3 重対角行列 T の各要素に関してミウラ変換、d L V v 型変換、逆ミウラ変換を行うことによって、前記対称 3 重対角行列 T を上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  にコレスキー分解するコレスキー分解部と、

前記上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  の各要素と、前記対称 3 重対角行列 T の固有値とを用いて固有ベクトルを算出して前記固有ベクトル記憶部に蓄積するベクトル算出部と、を備えた、請求項 4 記載の固有値分解装置。

10

## 【請求項 6】

前記コレスキー分解部は、複数のコレスキー分解手段を備え、

前記複数のコレスキー分解手段が、前記対称 3 重対角行列 T をコレスキー分解する処理を並列実行する、請求項 3 または請求項 5 記載の固有値分解装置。

## 【請求項 7】

前記ベクトル算出部は、複数のベクトル算出手段を備え、

前記複数のベクトル算出手段が、固有ベクトルを算出する処理を並列実行する、請求項 3、請求項 5 または請求項 6 記載の固有値分解装置。

## 【請求項 8】

前記固有値算出部は、複数の固有値算出手段を備え、

前記複数の固有値算出手段が、分割元の対称 3 重対角行列の固有値と、行列要素とを算出する処理を並列実行する、請求項 1 から請求項 7 のいずれか記載の固有値分解装置。

20

## 【請求項 9】

前記固有値分解部は、複数の固有値分解手段を備え、

前記複数の固有値分解手段が、対称 3 重対角行列に対して固有値分解を行う処理を並列実行する、請求項 1 から請求項 8 のいずれか記載の固有値分解装置。

## 【請求項 10】

対称行列 A が記憶される行列記憶部と、

前記対称行列 A を前記行列記憶部から読み出し、前記対称行列 A を 3 重対角化した前記対称 3 重対角行列 T を算出して前記対角行列記憶部に蓄積する対角化部と、をさらに備えた請求項 1 から請求項 9 のいずれか記載の固有値分解装置。

30

## 【請求項 11】

前記行列分割部は、対称 3 重対角行列を略半分の 2 個の対称 3 重対角行列に分割する、請求項 1 から請求項 10 のいずれか記載の固有値分解装置。

## 【請求項 12】

前記固有値算出部は、対称 3 重対角行列 T の全ての固有値を算出する、請求項 1 から請求項 11 のいずれか記載の固有値分解装置。

## 【請求項 13】

前記固有ベクトル算出部は、対称 3 重対角行列 T の全ての固有ベクトルを算出する、請求項 12 記載の固有値分解装置。

40

## 【請求項 14】

前記対称行列 A は、顔画像データを示すベクトルの平均から算出された共分散行列である、請求項 10 記載の固有値分解装置。

## 【請求項 15】

対称 3 重対角行列 T が記憶される対角行列記憶部と、行列分割部と、固有値分解部と、前記固有値分解部によって固有値分解された固有値と固有ベクトルからなる直交行列の一部の要素である行列要素とが記憶される固有値分解記憶部と、前記対称 3 重対角行列 T の固有値が記憶される固有値記憶部と、固有値算出部と、前記対称 3 重対角行列 T の固有ベクトルが記憶される固有ベクトル記憶部と、固有ベクトル算出部とを備えた固有値分解装置で用いられる固有値分解方法であって、

50

前記行列分割部が、前記対角行列記憶部から前記対称 3 重対角行列 T を読み出し、当該対称 3 重対角行列 T を 2 個の対称 3 重対角行列に分割して前記対角行列記憶部に蓄積し、その対称 3 重対角行列を 2 個の対称 3 重対角行列に分割して前記対角行列記憶部に蓄積する処理を、分割後の各対称 3 重対角行列があらかじめ決められた大きさ以下となるまで繰り返す行列分割ステップと、

前記固有値分解部が、前記あらかじめ決められた大きさ以下の各対称 3 重対角行列を前記対角行列記憶部から読み出し、前記各対称 3 重対角行列に対して固有値分解を行い、前記各対称 3 重対角行列の固有値と、前記各対称 3 重対角行列の固有ベクトルからなる直交行列の行列要素とを少なくとも算出し、当該固有値と当該行列要素とを前記固有値分解記憶部に蓄積する固有値分解ステップと、

前記固有値算出部が、各対称 3 重対角行列の固有値と、行列要素とを前記固有値分解記憶部から読み出し、前記固有値と、前記行列要素とから、分割元の対称 3 重対角行列の固有値と、分割元の対称 3 重対角行列の行列要素とを算出して前記固有値分解記憶部に蓄積し、その分割元の対称 3 重対角行列の固有値と、行列要素とを算出する処理を、対称 3 重対角行列 T の少なくとも 1 個の固有値を算出するまで繰り返し、前記対称 3 重対角行列 T の少なくとも 1 個の固有値を前記固有値記憶部に蓄積する固有値算出ステップと、

前記固有ベクトル算出部が、前記対角行列記憶部から前記対称 3 重対角行列 T を読み出し、前記固有値記憶部から前記対称 3 重対角行列 T の固有値を読み出し、前記対称 3 重対角行列 T とその固有値とから、ツイスト分解法を用いて前記対称 3 重対角行列 T の少なくとも 1 個の固有ベクトルを算出して前記固有ベクトル記憶部に蓄積する固有ベクトル算出ステップと、を備えた固有値分解方法。

【請求項 16】

コンピュータに、

対称 3 重対角行列 T が記憶される対角行列記憶部から前記対称 3 重対角行列 T を読み出し、当該対称 3 重対角行列 T を 2 個の対称 3 重対角行列に分割して前記対角行列記憶部に蓄積し、その対称 3 重対角行列を 2 個の対称 3 重対角行列に分割して前記対角行列記憶部に蓄積する処理を、分割後の各対称 3 重対角行列があらかじめ決められた大きさ以下となるまで繰り返す行列分割ステップと、

前記あらかじめ決められた大きさ以下の各対称 3 重対角行列を前記対角行列記憶部から読み出し、前記各対称 3 重対角行列に対して固有値分解を行い、前記各対称 3 重対角行列の固有値と、前記各対称 3 重対角行列の固有ベクトルからなる直交行列の行列要素とを少なくとも算出し、当該固有値と当該行列要素とを固有値分解記憶部に蓄積する固有値分解ステップと、

各対称 3 重対角行列の固有値と、行列要素とを前記固有値分解記憶部から読み出し、前記固有値と、前記行列要素とから、分割元の対称 3 重対角行列の固有値と、分割元の対称 3 重対角行列の行列要素とを算出して前記固有値分解記憶部に蓄積し、その分割元の対称 3 重対角行列の固有値と、行列要素とを算出する処理を、対称 3 重対角行列 T の少なくとも 1 個の固有値を算出するまで繰り返し、前記対称 3 重対角行列 T の少なくとも 1 個の固有値を固有値記憶部に蓄積する固有値算出ステップと、

前記対角行列記憶部から前記対称 3 重対角行列 T を読み出し、前記固有値記憶部から前記対称 3 重対角行列 T の固有値を読み出し、前記対称 3 重対角行列 T とその固有値とから、ツイスト分解法を用いて前記対称 3 重対角行列 T の少なくとも 1 個の固有ベクトルを算出して固有ベクトル記憶部に蓄積する固有ベクトル算出ステップと、を実行させるためのプログラム。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、固有値分解を行う固有値分解装置等に関する。

【背景技術】

【0002】

10

20

30

40

50

固有値分解は、例えば、分子軌道法による化学計算、統計計算、情報検索等の非常に広い分野で利用されている。また、固有値分解の方法も知られている（例えば、非特許文献1参照）。

【非特許文献1】J. J. M. Cuppen, 「A divide and conquer method for the symmetric tridiagonal eigenproblem」、Numerische Mathematik, Vol. 36, p. 177 - 195、1981年

【発明の開示】

【発明が解決しようとする課題】

【0003】

10

近年、これらの応用分野におけるデータ量の増大などに伴って、高速・高精度な固有値分解が求められている。また、固有値分解を並列処理することができる固有値分解装置等の開発が望まれていた。

【0004】

本発明は、上記状況の下になされたものであり、並列性に優れた高速、高精度な固有値分解を実行可能な固有値分解装置等を提供することを目的とする。

【課題を解決するための手段】

【0005】

20

上記目的を達成するため、本発明による固有値分解装置は、対称3重対角行列Tが記憶される対角行列記憶部と、前記対角行列記憶部から前記対称3重対角行列Tを読み出し、当該対称3重対角行列Tを2個の対称3重対角行列に分割して前記対角行列記憶部に蓄積し、その対称3重対角行列を2個の対称3重対角行列に分割して前記対角行列記憶部に蓄積する処理を、分割後の各対称3重対角行列があらかじめ決められた大きさ以下となるまで繰り返す行列分割部と、前記あらかじめ決められた大きさ以下の各対称3重対角行列の固有値と、当該各対称3重対角行列の固有ベクトルからなる直交行列の一部の要素である行列要素とが記憶される固有値分解記憶部と、前記あらかじめ決められた大きさ以下の各対称3重対角行列を前記対角行列記憶部から読み出し、前記各対称3重対角行列に対して固有値分解を行い、前記各対称3重対角行列の固有値と、前記各対称3重対角行列の固有ベクトルからなる直交行列の行列要素とを少なくとも算出し、当該固有値と当該行列要素とを前記固有値分解記憶部に蓄積する固有値分解部と、前記対称3重対角行列Tの固有値が記憶される固有値記憶部と、各対称3重対角行列の固有値と、行列要素とを前記固有値分解記憶部から読み出し、前記固有値と、前記行列要素とから、分割元の対称3重対角行列の固有値と、分割元の対称3重対角行列の行列要素とを算出して前記固有値分解記憶部に蓄積し、その分割元の対称3重対角行列の固有値と、行列要素とを算出する処理を、対称3重対角行列Tの少なくとも1個の固有値を算出するまで繰り返し、前記対称3重対角行列Tの少なくとも1個の固有値を前記固有値記憶部に蓄積する固有値算出部と、前記対称3重対角行列Tの固有ベクトルが記憶される固有ベクトル記憶部と、前記対角行列記憶部から前記対称3重対角行列Tを読み出し、前記固有値記憶部から前記対称3重対角行列Tの固有値を読み出し、前記対称3重対角行列Tとその固有値とから、ツイスト分解法を用いて前記対称3重対角行列Tの少なくとも1個の固有ベクトルを算出して前記固有ベクトル記憶部に蓄積する固有ベクトル算出部と、を備えたものである。

30

40

【0006】

このような構成により、まず固有値を算出し、その固有値に基づいてツイスト分解法を用いて固有ベクトルを算出することによって、高速で高精度な固有値分解を実現することができる。また、並列性にも優れている。さらに、全ての固有値や固有ベクトルを算出する必要がない場合には、必要な範囲で固有値や固有ベクトルを算出することができ、処理負荷を軽減することができる。

【0007】

また、本発明による固有値分解装置では、前記固有ベクトル算出部は、qd型ツイスト分解法により固有ベクトルを算出してもよい。

50

このような構成により、高速で高精度な固有値分解を実現することができる。また、並列性にも優れている。

【 0 0 0 8 】

また、本発明による固有値分解装置では、前記固有ベクトル算出部は、前記固有値記憶部から前記対称 3 重対角行列 T の固有値を読み出し、当該固有値のいずれかが負の値である場合に、前記対角行列記憶部から前記対称 3 重対角行列 T を読み出し、当該対称 3 重対角行列 T の固有ベクトルを変化させることなく、当該対称 3 重対角行列 T のすべての固有値が正の値となるように、当該対称 3 重対角行列 T を正定値化し、正定値化した後の固有値を前記固有値記憶部に蓄積し、正定値化した後の対称 3 重対角行列 T を前記対角行列記憶部に蓄積する正定値化部と、前記対角行列記憶部からすべての固有値が正の値である対称 3 重対角行列 T を読み出し、前記固有値記憶部から前記対称 3 重対角行列 T の正の値の固有値を読み出し、前記対称 3 重対角行列 T の各要素に関して q d 型変換を行うことによって、前記対称 3 重対角行列 T を上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  にコレスキー分解するコレスキー分解部と、前記上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  の各要素と、前記対称 3 重対角行列 T の正の値の固有値とを用いて固有ベクトルを算出して前記固有ベクトル記憶部に蓄積するベクトル算出部と、を備えてもよい。

10

【 0 0 0 9 】

このような構成により、負の値の固有値が含まれる場合には、対称 3 重対角行列 T を正定値化してから q d 型ツイスト分解法を実行することができる。また、L V 型ツイスト分解法よりも高速に計算することができる。

20

【 0 0 1 0 】

また、本発明による固有値分解装置では、前記固有ベクトル算出部は、L V 型ツイスト分解法により固有ベクトルを算してもよい。

このような構成により、数値安定的に固有ベクトルの算出を行うことができる。

【 0 0 1 1 】

また、本発明による固有値分解装置では、前記固有ベクトル算出部は、前記対角行列記憶部から前記対称 3 重対角行列 T を読み出し、前記固有値記憶部から前記対称 3 重対角行列 T の固有値を読み出し、前記対称 3 重対角行列 T の各要素に関してミウラ変換、d L V 型変換、逆ミウラ変換を行うことによって、前記対称 3 重対角行列 T を上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  にコレスキー分解するコレスキー分解部と、前記上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  の各要素と、前記対称 3 重対角行列 T の固有値とを用いて固有ベクトルを算出して前記固有ベクトル記憶部に蓄積するベクトル算出部と、を備えてもよい。

30

このような構成により、コレスキー分解の処理において、複数の補助変数を用いることによって、数値安定的に固有ベクトルの算出を行うことができる。

【 0 0 1 2 】

また、本発明による固有値分解装置では、前記コレスキー分解部は、複数のコレスキー分解手段を備え、前記複数のコレスキー分解手段が、前記対称 3 重対角行列 T をコレスキー分解する処理を並列実行してもよい。

40

このような構成により、コレスキー分解の処理を短時間で行うことができる。

【 0 0 1 3 】

また、本発明による固有値分解装置では、前記ベクトル算出部は、複数のベクトル算出手段を備え、前記複数のベクトル算出手段が、固有ベクトルを算出する処理を並列実行してもよい。

このような構成により、固有ベクトルを算出する処理を短時間で行うことができる。

【 0 0 1 4 】

また、本発明による固有値分解装置では、前記固有値算出部は、複数の固有値算出手段を備え、前記複数の固有値算出手段が、分割元の対称 3 重対角行列の固有値と、行列要素とを算出する処理を並列実行してもよい。

50

このような構成により、固有値を算出する処理を短時間で行うことができる。

【0015】

また、本発明による固有値分解装置では、前記固有値分解部は、複数の固有値分解手段を備え、前記複数の固有値分解手段が、対称3重対角行列に対して固有値分解を行う処理を並列実行してもよい。

このような構成により、固有値分解する処理を短時間で行うことができる。

【0016】

また、本発明による固有値分解装置では、対称行列Aが記憶される行列記憶部と、前記対称行列Aを前記行列記憶部から読み出し、前記対称行列Aを3重対角化した前記対称3重対角行列Tを算出して前記対角行列記憶部に蓄積する対角化部と、をさらに備えてもよい。

10

【0017】

このような構成により、任意の対称行列Aについて固有値を算出することができる。また、対称3重対角行列Tの固有ベクトルを用いることにより、対称行列Aの固有ベクトルを算出することもできる。

【0018】

また、本発明による固有値分解装置では、前記行列分割部は、対称3重対角行列を略半分の2個の対称3重対角行列に分割してもよい。

このような構成により、並列処理を適切に行うことができる。

【0019】

また、本発明による固有値分解装置では、前記固有値算出部は、対称3重対角行列Tの全ての固有値を算出してもよい。

20

【0020】

また、本発明による固有値分解装置では、前記固有ベクトル算出部は、対称3重対角行列Tの全ての固有ベクトルを算出してもよい。

【0021】

また、本発明による固有値分解装置では、前記対称行列Aは、顔画像データを示すベクトルの平均から算出された共分散行列であってもよい。

このような構成により、固有値分解装置を顔画像データの認識の処理に用いることができる。

30

【発明の効果】

【0022】

本発明による固有値分解装置等によれば、高速で高精度な固有値分解を行うことができる。また、並列性にも優れている。

【発明を実施するための最良の形態】

【0023】

以下、本発明による固有値分解装置について、実施の形態を用いて説明する。なお、以下の実施の形態において、同じ符号を付した構成要素及びステップは同一または相当するものであり、再度の説明を省略することがある。

【0024】

40

(実施の形態1)

本発明の実施の形態1による固有値分解装置について、図面を参照しながら説明する。

図1は、本実施の形態による固有値分解装置1の構成を示すブロック図である。図1において、本実施の形態による固有値分解装置1は、行列記憶部11と、対角化部12と、対角行列記憶部13と、行列分割部14と、固有値分解部15と、固有値分解記憶部16と、固有値算出部17と、固有値記憶部18と、固有ベクトル算出部19と、固有ベクトル記憶部20とを備える。

【0025】

行列記憶部11では、任意の対称行列Aが記憶される。この対称行列Aは、各要素が実数である実行列である。なお、対称行列Aが記憶されているとは、対称行列Aを示すデー

50





行列の固有値と、分割元の対称 3 重対角行列の行列要素とを算出して固有値分解記憶部 16 に蓄積する。固有値算出部 17 は、その分割元の対称 3 重対角行列の固有値と、行列要素とを算出する処理を、対称 3 重対角行列 T の固有値を算出するまで再帰的に繰り返す。そして、固有値算出部 17 は、算出した対称 3 重対角行列 T の固有値を固有値記憶部 18 に蓄積する。

【0032】

固有値記憶部 18 では、対称 3 重対角行列 T の固有値が記憶される。固有値記憶部 18 は、所定の記録媒体（例えば、半導体メモリや磁気ディスク、光ディスクなど）によって実現されうる。固有値記憶部 18 での記憶は、RAM 等における一時的な記憶でもよく、あるいは、長期的な記憶でもよい。

10

【0033】

固有ベクトル算出部 19 は、対角行列記憶部 13 から対称 3 重対角行列 T を読み出し、固有値記憶部 18 から対称 3 重対角行列 T の固有値を読み出す。そして、固有ベクトル算出部 19 は、対称 3 重対角行列 T とその固有値とから、ツイスト分解法を用いて対称 3 重対角行列 T の固有ベクトルを算出して固有ベクトル記憶部 20 に蓄積する。固有ベクトル算出部 19 は、LV 型ツイスト分解法により固有ベクトルを算出してもよく、あるいは、qd 型ツイスト分解法により固有ベクトルを算出してもよい。なお、具体的なツイスト分解の手法については後述する。

【0034】

固有ベクトル記憶部 20 では、対称 3 重対角行列 T の固有ベクトルが記憶される。固有ベクトル記憶部 20 は、所定の記録媒体（例えば、半導体メモリや磁気ディスク、光ディスクなど）によって実現されうる。固有ベクトル記憶部 20 での記憶は、RAM 等における一時的な記憶でもよく、あるいは、長期的な記憶でもよい。

20

【0035】

なお、行列記憶部 11、対角行列記憶部 13、固有値分解記憶部 16、固有値記憶部 18、固有ベクトル記憶部 20 の任意の 2 以上の記憶部は、同一の記録媒体によって実現されてもよく、あるいは、別々の記録媒体によって実現されてもよい。前者の場合には、例えば、対称行列 A の記憶されている領域が行列記憶部 11 となり、対称 3 重対角行列 T 等の記憶されている領域が対角行列記憶部 13 となる。

【0036】

また、行列記憶部 11、対角行列記憶部 13、固有値分解記憶部 16、固有値記憶部 18、固有ベクトル記憶部 20 の各記憶部は、2 以上の記録媒体から構成されてもよい。

30

【0037】

次に、本実施の形態による固有値分解装置 1 の動作について、図 2 のフローチャートを用いて説明する。

（ステップ S101）対角化部 12 は、行列記憶部 11 で記憶されている対称行列 A を読み出し、その行列 A を 3 重対角化して対称 3 重対角行列 T を算出して対角行列記憶部 13 に蓄積する。

【0038】

（ステップ S102）行列分割部 14、固有値分解部 15、固有値算出部 17 によって対称 3 重対角行列 T の固有値が算出され、固有値記憶部 18 に蓄積される。この処理の詳細については後述する。

40

【0039】

（ステップ S103）固有ベクトル算出部 19 は、対角行列記憶部 13 から対称 3 重対角行列 T を読み出し、固有値記憶部 18 から対称 3 重対角行列 T の固有値を読み出し、対称 3 重対角行列 T の固有ベクトルを算出して固有ベクトル記憶部 20 に蓄積する。この処理の詳細については後述する。

【0040】

このようにして、対称 3 重対角行列 T の固有値分解が終了する。ここで、対称行列 A の固有値は対称 3 重対角行列 T の固有値に等しいため、対称行列 A の固有値も算出されたこ

50

とになる。また、後述するように、所定の変換を行うことによって、対称行列 A の固有ベクトルも対称 3 重対角行列 T の固有ベクトルから容易に算出することができる。

【 0 0 4 1 】

次に、図 2 のフローチャートのステップ S 1 0 1 , S 1 0 2 の処理、すなわち、対称 3 重対角行列 T の固有値を算出するまでの処理について、より詳細に説明する。まず、図 2 のフローチャートのステップ S 1 0 2 の処理について、図 3 のフローチャートを用いて説明する。

【 0 0 4 2 】

(ステップ S 2 0 1) 行列分割部 1 4 は、対角行列記憶部 1 3 から対称 3 重対角行列 T を読み出し、その対称 3 重対角行列 T を 2 個の対称 3 重対角行列に分割して対角行列記憶部 1 3 に蓄積する。行列分割部 1 4 は、その対称 3 重対角行列を 2 個の対称 3 重対角行列に分割する処理を、分割後の各対称 3 重対角行列があらかじめ決められた大きさ以下となるまで繰り返す。この処理の詳細については後述する。

【 0 0 4 3 】

(ステップ S 2 0 2) 固有値分解部 1 5 は、対角行列記憶部 1 3 で記憶されているあらかじめ決められた大きさ以下の各対称 3 重対角行列について、固有値分解を行う。固有値分解部 1 5 は、固有値分解の結果である固有値と、固有ベクトルからなる直交行列の一部の要素である行列要素とを固有値分解記憶部 1 6 に蓄積する。

【 0 0 4 4 】

(ステップ S 2 0 3) 固有値算出部 1 7 は、対称 3 重対角行列の固有値と、行列要素とを固有値分解記憶部 1 6 から読み出し、分割元の対称 3 重対角行列の固有値と、分割元の対称 3 重対角行列の行列要素とを算出して固有値分解記憶部 1 6 に蓄積する。固有値算出部 1 7 は、その分割元の対称 3 重対角行列の固有値と、行列要素とを算出する処理を、対称 3 重対角行列 T の固有値を算出するまで繰り返し、対称 3 重対角行列 T の固有値を固有値記憶部 1 8 に蓄積する。このようにして、固有値を算出する処理が終了する。

【 0 0 4 5 】

[ 対称行列 A の対角化 ]

対称行列 A は、例えば、ハウスホルダー変換等を用いて、以下に示されるように 3 重対角化することができることが知られている。ここで、対称 3 重対角行列 T は、行の数と列の数とが一致する正方行列である。

【 0 0 4 6 】

【 数 1 】

$$P^T A P = T, \quad T = \begin{pmatrix} t_1 & t_2 & & & 0 \\ t_2 & t_3 & & & \\ & & \ddots & & \\ & & & \ddots & t_{2n-2} \\ 0 & & & t_{2n-2} & t_{2n-1} \end{pmatrix}$$

ただし、A は  $n \times n$  実対称行列、P は直交行列である。

【 0 0 4 7 】

したがって、対角化部 1 2 は、上記のようにして、行列記憶部 1 1 から対称行列 A を読み出し、対称 3 重対角行列 T を算出することができる。その算出された対称 3 重対角行列 T は、対角行列記憶部 1 3 で記憶される (ステップ S 1 0 1)。

【 0 0 4 8 】

[ 対称 3 重対角行列 T の分割 ]

対称 3 重対角行列 T を分割する処理について説明する。次式で示されるように、 $n \times n$  の対称 3 重対角行列 T が与えられた場合に、それらを 2 個の対称 3 重対角行列  $T_1, T_2$  と、その他の要素 (下記の と ) とに分けることができる。

【 0 0 4 9 】

10

20

30

40



るいは、 $n/2$ を下まわらない最小の整数にとることが好適である（この $n_1$ の値を用いて行列を2個の行列に分割する場合を、「行列を略半分の2個の行列に分割する」と呼ぶことにする）。

【0052】

【数4】

$$n_1 = \lfloor n/2 \rfloor$$

本実施の形態では、 $n_1$ を上式のようにとるものとする。なお、 $n_1$ の値は、前述のように、 $1 < n_1 < n$ の範囲内で任意であることは言うまでもない。

【0053】

10

したがって、行列分割部14は、上述のようにして、対角行列記憶部13が記憶している対称3重対角行列 $T$ を読み出し、その対称3重対角行列 $T$ を2個の対称3重対角行列 $T_1$ 、 $T_2$ と、2個の値、 $\alpha$ 、 $\beta$ に分割することができ、その分割の処理を繰り返すことができる。行列分割部14は、分割後の2個の対称3重対角行列 $T_1$ 、 $T_2$ と、2個の値、 $\alpha$ 、 $\beta$ とを、対角行列記憶部13に蓄積する。なお、2個の値、 $\alpha$ 、 $\beta$ は、どの分割で発生した値であるのかがわかるように対角行列記憶部13で記憶されることが好適である。図4は、図3のフローチャートのステップS201における行列分割部14による行列を分割する処理を示すフローチャートである。

【0054】

（ステップS301）行列分割部14は、カウンタ $I$ を「1」に設定する。

20

（ステップS302）行列分割部14は、対角行列記憶部13から $I$ 番目の分割をしていない対称3重対角行列を読み出し、その対称3重対角行列を2個の対称3重対角行列と、その他の要素とに分割する。そして、行列分割部14は、分割した2個の対称3重対角行列と、その他の要素とを対角行列記憶部13に蓄積する。

【0055】

（ステップS303）行列分割部14は、 $I$ 番目の分割を行っていない対称3重対角行列が、対角行列記憶部13で記憶されているかどうか判断する。そして、 $I$ 番目の分割を行っていない対称3重対角行列が、対角行列記憶部13で記憶されている場合には、ステップS302に戻り、そうでない場合には、ステップS304に進む。

【0056】

30

（ステップS304）行列分割部14は、 $I$ 番目の分割を行った対称3重対角行列の大きさがあらかじめ決められている大きさ以下かどうか判断する。行列分割部14は、例えば、目的とする行列の大きさ（例えば、 $25 \times 25$ など）の記憶されている図示しない記録媒体から、その目的とする行列の大きさを読み出し、対角行列記憶部13で記憶されている、 $I$ 番目の分割後の対称3重対角行列がその大きさ以下であるかどうかを判断してもよい。そして、 $I$ 番目の分割を行った対称3重対角行列の大きさがあらかじめ決められている大きさ以下である場合には、対称3重対角行列を分割する処理は終了となり、そうでない場合には、ステップS305に進む。

（ステップS305）行列分割部14は、カウンタ $I$ を1だけインクリメントする。そして、ステップS302に戻る。

40

【0057】

なお、この図4のフローチャートでは、上述のように、行列分割部14が各行列を、略半分の2個の行列に分割するため、 $I$ 番目の分割後の各対称3重対角行列の大きさがほとんど同じである場合について説明したが、行列分割部14が各行列を、略半分の2個の行列に分割しない場合には、行列分割部14は、分割後の各行列があらかじめ決められた大きさ以下となるように、分割を繰り返すものとする。

【0058】

また、図4のフローチャートでは、ステップS304において、行列分割部14が、対角行列記憶部13で記憶されている分割後の行列の大きさをあらかじめ決められた大きさと比較する処理を実行する場合について説明したが、これは一例であって、行列分割部1

50

4 は、ステップ S 3 0 4 において、それ以外の処理を行ってもよい。例えば、上述のように、行列分割部 1 4 が各行列を、略半分の 2 個の行列に分割する場合には、元の対称 3 重対角行列 T の大きさを知ることができれば、何番目の分割で目的とする行列の大きさとなるのかを知ることができる。したがって、N 番目 (N は 1 以上の整数) の分割で目的とする行列の大きさとなる場合には、ステップ S 3 0 4 において、I が N であるかどうかを比較し、N でなければステップ S 3 0 5 に進み、N であれば一連の処理を終了するようにしてもよい。

#### 【 0 0 5 9 】

図 5 は、行列の分割について説明するための図である。まず、行列分割部 1 4 は、1 番目の分割として、対称 3 重対角行列 T を、対称 3 重対角行列  $T_1$  と、対称 3 重対角行列  $T_2$  とに分割する (ステップ S 3 0 1, S 3 0 2)。対称 3 重対角行列 T は、1 個しかないので、行列分割部 1 4 は、1 番目の分割をしていない行列がないと判断する (ステップ S 3 0 3)。また、対称 3 重対角行列  $T_1$  等はあらかじめ決められた大きさ以下の行列でないとすると (ステップ S 3 0 4)、行列分割部 1 4 は、2 番目の分割として、対称 3 重対角行列  $T_1$  を、対称 3 重対角行列  $T_{11}$  と、対称 3 重対角行列  $T_{12}$  とに分割する (ステップ S 3 0 5, S 3 0 2)。この場合には、2 番目の分割を行っていない対称 3 重対角行列  $T_2$  が存在するため、行列分割部 1 4 は、対称 3 重対角行列  $T_2$  も、対称 3 重対角行列  $T_{21}$  と、対称 3 重対角行列  $T_{22}$  とに分割する (ステップ S 3 0 3, S 3 0 2)。このようにして、分割後の各対称 3 重対角行列が目的とする大きさ以下になるまで、行列を分割する処理が繰り返される。なお、図 5 では、対称 3 重対角行列以外の要素 (上述の、 ) については省略している。また、図 5 において、各対称 3 重対角行列は、正方行列である。

#### 【 0 0 6 0 】

[ 分割された行列の固有値分解 ]

行列  $T_i$  が対称行列であるとする、行列  $T_i$  の固有値分解は、次のようになる。

#### 【 数 5 】

$$T_i = Q_i D_i Q_i^T$$

#### 【 0 0 6 1 】

ここで、 $D_i$  は、行列  $T_i$  と同じ大きさの対角行列である。 $D_i$  の各対角成分は  $T_i$  の固有値である。また、 $Q_i$  は、行列  $T_i$  の固有ベクトルからなる直交行列である。

#### 【 0 0 6 2 】

したがって、固有値分解部 1 5 は、対角行列記憶部 1 3 から分割後の各対称 3 重対角行列を読み出し、上記のようにして、固有値分解を行う (ステップ S 2 0 2)。固有値分解の方法として、例えば、2 分法と逆反復法を組み合わせた方法、MR<sup>3</sup>法、QR法等を用いてもよいことは前述の通りである。図 5 で示されるように対称 3 重対角行列の分割が行われた場合には、固有値分解部 1 5 は、各対称 3 重対角行列  $T_{111}$ ,  $T_{112}$ ,  $T_{121}$ ,  $T_{122}$ , ...,  $T_{222}$  に対して固有値分解を行うことになる。なお、固有値分解部 1 5 は、固有値分解の結果得られた各固有値と、固有値分解の結果得られた直交行列の一部の要素である行列要素とを固有値分解記憶部 1 6 に蓄積する。ここで、行列要素が、直交行列のどの要素を含むのかについては後述する。なお、固有値分解部 1 5 が固有値と行列要素とを固有値分解記憶部 1 6 に蓄積するのではなく、固有値と固有ベクトルからなる直交行列とを固有値分解記憶部 1 6 に蓄積してもよい。その場合でも、固有ベクトルからなる直交行列を蓄積することによって、少なくとも、行列要素が蓄積されたことになる。

#### 【 0 0 6 3 】

なお、後述するように、固有値を算出するときに用いるために必要なのは、直交行列  $Q_i$  の全体ではなく、その直交行列  $Q_i$  の第 1 行と最終行である。したがって、この固有値分解において、直交行列  $Q_i$  の全体を求めるのではなく、直交行列  $Q_i$  の第 1 行と最終行を求めるようにしてもよい。そのためには、例えば、LAPACK 3.0 で提供されてい

るQR法の固有値分解ルーチンであるDSTEQRにおいて、直交行列 $Q_i$ の算出における初期値として、単位行列 $I$ を用いる代わりに、次式で示される行列 $Y$ を用いてもよい。

【数6】

$$Y = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & 0 & 1 \end{pmatrix}$$

ただし、 $Q_i$ は $n \times n$ の直交行列であり、 $Y$ は $2 \times n$ の行列である。

【0064】

このようにすることで、DSTEQRで直交行列 $Q_i$ の第1行と最終行を求めるための計算領域を削減することができる。直交行列 $Q_i$ そのものを算出する場合には、 $O(n^2)$ の作業領域が必要であるが、上記の行列 $Y$ を用いて直交行列 $Q_i$ の第1行と最終行を算出する場合には、 $O(n)$ の作業領域でよいことになる。その結果、計算量も少なくなり、固有値分解の処理速度も向上する。このように、固有値分解部15は、分割後の対称3重対角行列に対して、直交行列 $Q_i$ そのものを算出する固有値分解を行ってもよく、あるいは、直交行列の一部の成分（ここでは、第1行と最終行）を算出する処理を行ってもよい。この後者の処理も固有値分解と呼ぶことにする。このように、固有値分解部15は、あらかじめ決められた大きさ以下の各対称3重対角行列に対して固有値分解を行い、各対称3重対角行列の固有値と、各対称3重対角行列の固有ベクトルからなる直交行列の行列要素（ここでは、第1行と最終行）とを少なくとも算出し、その固有値と、その行列要素とを固有値分解記憶部16に蓄積してもよい。

10

20

【0065】

[固有値の算出]

まず、図6で示されるように、分割された2個の対称3重対角行列 $T_1, T_2$ から、分割元の対称3重対角行列 $T_0$ の固有値等を算出する処理について説明する。対称3重対角行列 $T_1, T_2$ は、次のように固有値分解されていたとする。ここで、対称3重対角行列 $T_1, T_2$ は、両者共に正方行列であるとする。

【数7】

$$T_1 = Q_1 D_1 Q_1^T$$

$$T_2 = Q_2 D_2 Q_2^T$$

30

【0066】

この場合に、分割元の対称3重対角行列 $T_0$ は、次のようになる。

【数8】

$$T_0 = Q_{12} (D_{12} + \rho z z^T) Q_{12}^T$$

$$Q_{12} \equiv \begin{pmatrix} Q_1 & 0 \\ 0 & Q_2 \end{pmatrix}, \quad D_{12} \equiv \begin{pmatrix} D_1 & 0 \\ 0 & D_2 \end{pmatrix}$$

$$z \equiv \frac{1}{\sqrt{1+\theta^{-2}}} \begin{pmatrix} \mathbf{1}_1^T \\ \theta^{-1} \mathbf{f}_2^T \end{pmatrix}, \quad \rho \equiv \beta(\theta + \theta^{-1})$$

40

ただし、 $\mathbf{1}_1$ は $Q_1$ の最終行であり、 $\mathbf{f}_2$ は $Q_2$ の第1行である。

【0067】

上記の式において、 $\rho$ 、 $z$ は、対称3重対角行列の分割の処理において説明した値である。ここで、行列 $(D_{12} + z z^T)$ の固有値分解が得られれば、行列 $T_0$ の固有値分解が得られたことになる。なぜなら、行列 $(D_{12} + z z^T)$ の固有値分解 $D_{12} + z z^T = Q D Q^T$ が求まると、次式のようになり、行列 $T_0$ の固有値分解が完了するからである。

【数9】

$$T_0 = Q_{12} (Q D Q^T) Q_{12}^T = (Q_{12} Q) D (Q_{12} Q)^T = Q_0 D Q_0^T$$

50

## 【 0 0 6 8 】

次に、行列  $(D_{12} + \rho \mathbf{z} \mathbf{z}^T)$  の固有値を算出する方法について説明する。行列  $(D_{12} + \rho \mathbf{z} \mathbf{z}^T)$  の固有値を  $\lambda_i$ 、固有ベクトルを  $\mathbf{q}_i$  とすると、

## 【 数 1 0 】

$$(D_{12} + \rho \mathbf{z} \mathbf{z}^T) \mathbf{q}_i = \lambda_i \mathbf{q}_i$$

となる。

## 【 0 0 6 9 】

(1)  $\mathbf{z}$  の成分に 0 が含まれず、 $D_{12}$  の対角成分に同じ値が含まれない場合

まず、 $\mathbf{z}$  の成分に 0 が含まれず、 $D_{12}$  の対角成分に同じ値が含まれない場合について説明する。その場合には、上記の式を、次に示すように順次、変形することができる。 10

## 【 数 1 1 】

$$(D_{12} - \lambda_i I) \mathbf{q}_i = -\rho (\mathbf{z}^T \mathbf{q}_i) \mathbf{z}$$

$$\mathbf{z}^T \mathbf{q}_i = -\rho (\mathbf{z}^T \mathbf{q}_i) \mathbf{z}^T (D_{12} - \lambda_i I)^{-1} \mathbf{z}$$

$$1 + \rho \mathbf{z}^T (D_{12} - \lambda_i I)^{-1} \mathbf{z} = 0$$

## 【 0 0 7 0 】

ここで、

## 【 数 1 2 】

$$D_{12} = \text{diag}(d_1, d_2, \dots, d_n)$$

20

$$\mathbf{z} = (z_1, z_2, \dots, z_n)^T$$

とすると、次の固有方程式を得ることができる。固有値  $\lambda_i$  ( $i = 1, 2, \dots, n$ ) は、その固有方程式  $n$  個の解である。下記の  $f(\lambda)$  は、単調増加であり、かつ  $f(\lambda) = 0$  の解の範囲が既知であるため、ニュートン法を用いて計算することができる。

## 【 数 1 3 】

$$f(\lambda) = 1 + \rho \sum_{j=1}^n \frac{z_j^2}{d_j - \lambda} = 0 \quad (\text{式 1})$$

## 【 0 0 7 1 】

30

なお、 $D = \text{diag}(d_1, d_2, \dots, d_n)$  となる。行列  $Q$  は、行列  $(D_{12} + \rho \mathbf{z} \mathbf{z}^T)$  の固有ベクトルからなる直交行列であるため、 $Q = (\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_n)$  となる。したがって、固有ベクトル  $\mathbf{q}_i$  を求めると、行列  $Q$  を求めることができる。なお、固有ベクトル  $\mathbf{q}_i$  は、次式で与えられる。

## 【 数 1 4 】

$$\mathbf{q}_i = \frac{\left( \frac{z_1}{d_1 - \lambda_i}, \frac{z_2}{d_2 - \lambda_i}, \dots, \frac{z_n}{d_n - \lambda_i} \right)^T}{\sqrt{\sum_{k=1}^n \frac{z_k^2}{(d_k - \lambda_i)^2}}} \quad (\text{式 2})$$

40

## 【 0 0 7 2 】

(2)  $\mathbf{z}$  の成分に 0 が含まれる場合

$\mathbf{z}$  の成分に 0 が含まれる場合には、行列の次数を下げることができ、その次数を下けた行列を用いて固有値、及び固有ベクトルを求めることができる。この行列の次数を下げる操作をデフレーションと呼ぶ。

## 【 0 0 7 3 】

$z_j = 0$  の場合には、次式で示すように、 $(d_j, \mathbf{e}_j)$  が一組の固有値と固有ベクトルとなる。ただし、 $\mathbf{e}_j$  は、 $j$  番目の成分が 1 であり、その他の成分が 0 である単位ベクトルである。

50

【数 1 5】

$$((D_{12} + \rho \mathbf{z} \mathbf{z}^T) - d_j I) \mathbf{e}_j = (D_{12} - d_j I) \mathbf{e}_j + \rho \mathbf{z} (\mathbf{z}^T \mathbf{e}_j) = 0$$

【0 0 7 4】

また、行列  $(D_{12} + \rho \mathbf{z} \mathbf{z}^T)$  の固有方程式は、次式のようにになる。

【数 1 6】

$$|D_{12} + \rho \mathbf{z} \mathbf{z}^T - \lambda_i I_n| = (d_j - \lambda_i) |\hat{D}_{12} + \rho \hat{\mathbf{z}} \hat{\mathbf{z}}^T - \lambda_i I_{n-1}|$$

ただし、 $\hat{D}_{12} + \rho \hat{\mathbf{z}} \hat{\mathbf{z}}^T$  は、 $D_{12} + \rho \mathbf{z} \mathbf{z}^T$  の第  $j$  行、第  $j$  列を除いた小行列である。

10

また、 $I_n$  は、 $n$  次の単位行列である。

【0 0 7 5】

したがって、 $z_j = 0$  の場合の固有値は、 $d_j$  と、行列  $(D_{12} + \rho \mathbf{z} \mathbf{z}^T)$  の第  $j$  行、第  $j$  列を除いた小行列について、上記式 1 の固有方程式を解いた値である。すなわち、固有値は、 $d_j$  と、次の固有方程式を解いた値となる。なお、次の固有方程式を解いた固有値を、固有値  $\lambda_i$  ( $i = 1, 2, \dots, j-1, j+1, \dots, n$ ) とする。

【数 1 7】

$$f(\lambda) = 1 + \rho \sum_{\substack{k=1 \\ k \neq j}}^n \frac{z_k^2}{d_k - \lambda} = 0$$

20

【0 0 7 6】

固有ベクトルは、行列  $(D_{12} + \rho \mathbf{z} \mathbf{z}^T)$  の第  $j$  行、第  $j$  列を除いた小行列について上記式 2 で算出した  $n-1$  次元の固有ベクトルの第  $(j-1)$  行と、第  $j$  行との間に 0 を挿入した次式の  $n$  次元のベクトルと、上述の  $\mathbf{e}_j$  となる。

【数 1 8】

$$\mathbf{q}_i = \frac{\left( \frac{z_1}{d_1 - \lambda_i}, \frac{z_2}{d_2 - \lambda_i}, \dots, \frac{z_{j-1}}{d_{j-1} - \lambda_i}, 0, \frac{z_j}{d_j - \lambda_i}, \dots, \frac{z_n}{d_n - \lambda_i} \right)^T}{\sqrt{\sum_{\substack{k=1 \\ k \neq j}}^n \frac{z_k^2}{(d_k - \lambda_i)^2}} \quad (i \neq j)$$

30

$$\mathbf{q}_j = \mathbf{e}_j$$

【0 0 7 7】

このように、 $z$  の成分に 0 が含まれる場合には、行列  $(D_{12} + \rho \mathbf{z} \mathbf{z}^T)$  よりも次元の小さい行列について固有値分解を行うことになるため、計算を高速化することが可能であり、結果として、計算時間が短縮される。

【0 0 7 8】

(3)  $D_{12}$  の対角成分に同じ値が含まれる場合

40

$d_{j+1} = d_{j+2} = \dots = d_k$  の場合にも、上記説明と同様のデフレーションを行うことによって、固有値と固有ベクトルとを算出することができる。

【0 0 7 9】

【数 1 9】

$$\hat{\mathbf{z}} = (z_{j+1}, z_{j+2}, \dots, z_k)^T$$

に対して、次の 2 個の直交行列を考える。



【数 2 0】

$$\hat{Q} \hat{\mathbf{z}} = (*, 0, 0, \dots, 0)^T$$

$$\tilde{Q} = \begin{pmatrix} I_j & & \\ & \hat{Q} & \\ & & I_{n-k} \end{pmatrix}$$

ただし、 $\hat{Q} \in \mathbf{R}^{(k-j) \times (k-j)}$ である。

また、\*は所定の実数である。

【0 0 8 0】

10

すると、 $d_{j+1} = d_{j+2} = \dots = d_k$ より、

【数 2 1】

$$\tilde{Q} D_{12} \tilde{Q}^T = D_{12}$$

$$\tilde{Q} \mathbf{z} = (z_1, z_2, \dots, z_j, *, 0, \dots, 0, z_{k+1}, \dots, z_n)^T = \tilde{\mathbf{z}}$$

となる。したがって、次式が得られる。

【数 2 2】

$$D_{12} + \rho \mathbf{z} \mathbf{z}^T = \tilde{Q}^T (D_{12} + \rho \tilde{\mathbf{z}} \tilde{\mathbf{z}}^T) \tilde{Q}$$

20

【0 0 8 1】

ここで、

【数 2 3】

$$D_{12} + \rho \tilde{\mathbf{z}} \tilde{\mathbf{z}}^T$$

の固有値と固有ベクトルとは、上記の  $\mathbf{z}$  の成分に 0 が含まれるデフレーションの場合と同様にして求めることができる。また、

【数 2 4】

$$\hat{Q}$$

30

は、符号の任意性を除いて一意に定まる。ここで、その算出方法について説明する。まず、次式を満たす直交行列について考える。

【0 0 8 2】

【数 2 5】

$$\hat{Q}_{j+1,j} \cdot (z_{j+1}, \dots, z_i, \dots, z_k)^T = (*, z_{j+2}, \dots, z_{i-1}, 0, z_{i+1}, \dots, z_k)^T$$

この直交行列を次式のようにとると、上記式を満たすことになる。

【数 2 6】

$$\hat{Q}_{j+1,j} = \begin{pmatrix} \frac{z_{j+1}}{\sqrt{z_{j+1}^2 + z_i^2}} & & \frac{z_i}{\sqrt{z_{j+1}^2 + z_i^2}} & & \\ & I_{i-j-2} & & & \\ \frac{-z_j}{\sqrt{z_{j+1}^2 + z_i^2}} & & \frac{z_{j+1}}{\sqrt{z_{j+1}^2 + z_i^2}} & & \\ & & & & I_{k-i} \end{pmatrix}$$

40

【0 0 8 3】

ここで、上記の式の \* の値は、次式から求まる。

【数 2 7】

$$\hat{Q}_{j+1,j} \cdot (z_{j+1}, \dots, z_i, \dots, z_k)^T = (\sqrt{z_{j+1}^2 + z_i^2}, z_{j+2}, \dots, z_{i-1}, 0, z_{i+1}, \dots, z_k)^T$$

【0 0 8 4】

したがって、

【数 2 8】

$$\hat{Q} = \hat{Q}_{j+1,j+2} \cdot \hat{Q}_{j+1,j+3} \cdots \hat{Q}_{j+1,k}$$

となる。その結果、

【数 2 9】

$$\tilde{Q}$$

も、符号の任意性を除いて求めることができる。このようにして、 $d_{j+1} = d_{j+2} = \dots = d_k$  の場合にも行列  $(D_{1,2} + z z^T)$  の固有値と固有ベクトルとを求めることができる。すなわち、固有値は、 $d_{j+2}, d_{j+3}, \dots, d_k$  と、次の固有方程式を解いた値となる。なお、次の固有方程式を解いた固有値を、固有値  $\lambda_i$  ( $i = 1, 2, \dots, j+1, k+1, \dots, n$ ) とする。

【数 3 0】

$$f(\lambda) = 1 + \rho \sum_{\substack{m=1 \\ m \neq j+2, \dots, k}}^n \frac{z_m^2}{d_m - \lambda} = 0$$

【0 0 8 5】

固有ベクトルは、次のようになる。

【数 3 1】

$$q_i = \begin{cases} \tilde{Q}^T \frac{\left( \frac{\tilde{z}_1}{d_1 - \lambda_i}, \frac{\tilde{z}_2}{d_2 - \lambda_i}, \dots, \frac{\tilde{z}_n}{d_n - \lambda_i} \right)^T}{\sqrt{\sum_{k=1}^n \frac{\tilde{z}_k^2}{(d_k - \lambda_i)^2}}} & (1 \leq i \leq j+1, k+1 \leq i \leq n) \\ \tilde{Q}^T e_i & (j+2 \leq i \leq k) \end{cases}$$

【0 0 8 6】

なお、上記(2)  $z$  の成分に 0 が含まれる場合、(3)  $D_{1,2}$  の対角成分に同じ値が含まれる場合において、 $z$  の成分が 0 であるかどうかの判断や、 $D_{1,2}$  の対角成分に同じ値が含まれているかどうかの判断は、微少な誤差を考慮して行ってもよい。例えば、次式が満たされる場合には、 $z$  の成分が 0 であると判断してもよく、また、 $D_{1,2}$  の対角成分が同じ値であると判断してもよい。

【数 3 2】

$$|z_i| < TOL_1$$

$$|d_{j+1} - d_k| < TOL_2$$

ただし、 $TOL_1 \equiv \max(z_1, z_2, \dots, z_n) \times \epsilon$ 、 $TOL_2 \equiv \max(d_1, d_2, \dots, d_n) \times \epsilon$  である。

【0 0 8 7】

なお、 $\epsilon$  は、マシン・イプシロンに定数(例えば、2, 4, 8 などの実数であり、計算誤差を少なくするための値を選択することが好適である)を掛けた値を用いてもよく、他の適切な値であってもよい。

【0 0 8 8】

このようにして、(1)  $z$  の成分に 0 が含まれず、 $D_{1,2}$  の対角成分に同じ値が含まれない場合、(2)  $z$  の成分に 0 が含まれる場合、(3)  $D_{1,2}$  の対角成分に同じ値が含ま

10

20

30

40

50

れる場合のそれぞれについて、固有値分解  $D_{12} + z z^T = Q D Q^T$  が求まることになる。したがって、行列  $Q$  と行列  $Q_{12}$  から行列  $Q_0$  を求めることができ、行列  $D$  も求まっているため、分割元の対称三重対角行列  $T_0$  の固有値分解を行うことができる。このように、対象となる行列を2個の副行列に分割し、その分割した後の各副行列について固有値分解を行い、その後各副行列の固有値分解を用いて分割元の行列の固有値分解を行う方法は、分割統治法 (Divide and Conquer: D&C) と呼ばれている。

【0089】

分割統治法では、固有ベクトルまで算出することになり、その固有ベクトルを算出する処理において行列の計算をしなければならないため、非常に負荷の大きい処理となる。分割統治法において固有値分解をする場合には、例えば、計算時間の95%程度が固有ベクトルを算出するベクトル更新の処理 (例えば、上述のように、行列  $Q$  と行列  $Q_{12}$  から行列  $Q_0$  を算出するために行列を掛け合わせる処理) に費やされることもありうる。

10

【0090】

一方、分割元の行列  $T_0$  の固有値のみを算出する場合には、上記処理を簡略化することができる。次に、その方法について説明する。上記の結果より、次のようになる。

【0091】

【数33】

$$\mathbf{f} = \mathbf{f}_{12} Q \quad (\text{式3})$$

$$\mathbf{l} = \mathbf{l}_{12} Q \quad (\text{式4})$$

ただし、 $\mathbf{f}$  は  $Q_0$  の第1行であり、 $\mathbf{l}$  は  $Q_0$  の最終行である。

20

また、 $\mathbf{f}_{12}$  は  $Q_{12}$  の第1行であり、 $\mathbf{f}_{12} = (\mathbf{f}_1, 0, \dots, 0)$  となる。

また、 $\mathbf{l}_{12}$  は  $Q_{12}$  の最終行であり、 $\mathbf{l}_{12} = (0, \dots, 0, \mathbf{l}_2)$  となる。

また、 $\mathbf{f}_1$  は  $Q_1$  の第1行であり、 $\mathbf{l}_2$  は  $Q_2$  の最終行である。

【0092】

ここで、上記のように、 $\mathbf{f}$  は行列  $T_0$  を固有値分解した結果の直交行列の最初の行であり、 $\mathbf{l}$  は行列  $T_0$  を固有値分解した結果の直交行列の最後の行である。この  $\mathbf{f}$  と  $\mathbf{l}$ 、すなわち、直交行列の最初の行と、最後の行とが行列要素となる。なお、直交行列  $Q_{12}$  について、後述する列の入れ替えを行った場合には、 $\mathbf{f}_{12}$ 、 $\mathbf{l}_{12}$  はそれぞれ、その列を入れ替えた後の直交行列  $Q_{12}$  の第1行、最終行となる。

30

【0093】

上記の結果から、図6で示されるように、行列  $T_1$ 、 $T_2$  から分割元の行列  $T_0$  を構成する場合に、行列  $T_0$  について、

【数34】

$$\lambda_i, \mathbf{f}, \mathbf{l}$$

をそれぞれ算出することができる。このような処理を繰り返すことにより、最終的に、対称行列  $A$  を三重対角化した対称三重対角行列  $T$  の固有値を算出することができる。

40

【0094】

なお、上記の(3)  $D_{12}$  の対角成分に同じ値が含まれる場合には、式3、式4が次のようになる。

【数35】

$$\mathbf{f} = \mathbf{f}_{12} Q = \mathbf{f}_{12} \tilde{Q}^T \bar{Q}$$

$$\mathbf{l} = \mathbf{l}_{12} Q = \mathbf{l}_{12} \tilde{Q}^T \bar{Q}$$

ただし、 $\bar{Q} = \tilde{Q} Q$  である。

【0095】

ここで、上記式を計算する場合に、

50

【数 3 6】

$$f_{12} \tilde{Q}^T, l_{12} \tilde{Q}^T$$

を先に計算して、その後

【数 3 7】

$$\bar{Q}$$

を掛けた方が、計算量が少なくなるため、その順番で計算するようにしてもよい。

【0096】

したがって、固有値算出部 17 は、上述のようにして、固有値分解記憶部 16 から固有値と行列要素とを読み出し、対角行列記憶部 13 から行列の分割時に発生したその他の要素に関する、 $\tilde{Q}$  を読み出し、それらを用いることによって、分割元の対称 3 重対角行列の固有値と行列要素とを算出する。そして、それらを算出する処理を繰り返すことによって、最終的に対称 3 重対角行列 T の固有値を算出する。図 7 は、図 3 のフローチャートのステップ S 203 における固有値算出部 17 が固有値を算出する処理を示すフローチャートである。

10

【0097】

(ステップ S 401) 固有値算出部 17 は、カウンタ J を「1」に設定する。

(ステップ S 402) 固有値算出部 17 は、J 番目の固有値の算出は最後の固有値の算出であるかどうか判断する。ここで、最後の固有値の算出とは、対称 3 重対角行列 T の固有値を算出することである。そして、最後の固有値の算出である場合には、ステップ S 406 に進み、そうでない場合には、ステップ S 403 に進む。

20

【0098】

(ステップ S 403) 固有値算出部 17 は、分割元の対称 3 重対角行列の固有値等を算出する。この処理の詳細については後述する。

(ステップ S 404) 固有値算出部 17 は、J 番目の固有値の算出において、分割元の全ての対称 3 重対角行列の固有値等を算出したかどうか判断する。そして、J 番目の固有値の算出において、分割元の全ての対称 3 重対角行列の固有値等を算出した場合には、ステップ S 405 に進み、そうでない場合には、ステップ S 403 に戻る。

【0099】

(ステップ S 405) 固有値算出部 17 は、カウンタ J を 1 だけインクリメントする。そして、ステップ S 402 に戻る。

30

(ステップ S 406) 固有値算出部 17 は、対称 3 重対角行列 T の固有値を算出し、その算出した固有値を固有値記憶部 18 に蓄積する。このようにして、対称 3 重対角行列 T の固有値を算出する一連の処理は終了となる。

【0100】

図 8 は、図 7 のフローチャートにおけるステップ S 403 の詳細な処理を示すフローチャートである。なお、この図 8 のフローチャートの処理を実行する前に、対角行列  $D_{12}$  の対角成分が昇順、あるいは降順となるように並べ替えておくことが好適である。なお、その場合には、直交行列  $Q_{12}$  の列、及び  $z$  の成分も、対角行列  $D_{12}$  の並び替えと同様に並び替えるものとする。具体的には、対角行列  $D_{12}$  の第 i 成分と、第 j 成分とを入れ替えた場合には、直交行列  $Q_{12}$  の第 i 列と、第 j 列とを入れ替え、 $z$  の第 i 行と、第 j 行とを入れ替えればよい。

40

【0101】

(ステップ S 501) 固有値算出部 17 は、分割元の行列の固有値を、式 1 等を用いて算出する。そして、固有値算出部 17 は、算出した固有値を固有値分解記憶部 16 に蓄積する。

【0102】

(ステップ S 502) 固有値算出部 17 は、ステップ S 501 で算出した固有値を用いて、式 2 等から  $q_i$  を算出する。この  $q_i$  を算出することにより、 $q_i$  を列ベクトルに有

50

する行列  $Q$  を算出したことになる。固有値算出部 17 は、算出した行列  $Q$  を図示しない記録媒体において一次記憶してもよい。

【0103】

(ステップ S503) 固有値算出部 17 は、ステップ S502 で算出した行列  $Q$  を用いて、分割元の行列に関する行列要素  $f$ 、 $l$  を算出する。この行列要素は、式 3、式 4 で示されるものである。そして、固有値算出部 17 は、算出した分割元の行列要素を固有値分解記憶部 16 に蓄積する。このようにしてステップ S403 の処理は終了となる。

【0104】

なお、図 8 のフローチャートにおいて、 $z$  の成分に 0 が含まれず、 $D_{12}$  の対角成分に同じ値が含まれない場合には、上記 (1) で説明したようにして、固有値、固有ベクトルからなる行列  $Q$  を算出することができる。また、 $z$  の成分に 0 が含まれ、 $D_{12}$  の対角成分に同じ値が含まれない場合には、上記 (2) で説明したようにして、固有値、固有ベクトルからなる行列  $Q$  を算出することができる。また、 $D_{12}$  の対角成分に同じ値が含まれる場合には、 $z$  の成分に 0 が含まれるかどうかに関わらず、上記 (3) で説明したようにして、固有値、固有ベクトルからなる行列  $Q$  を算出することができる。上記 (3) で説明したようにして固有値等を算出する場合には、前述のように、行列  $Q$  を算出するのではなく、行列  $Q$  を構成する行列を直接  $f_{12}$  等に作用させることによって、 $f$ 、 $l$  を算出するようにしてもよい。

10

【0105】

また、図 8 のフローチャートでは、行列  $Q$  から、行列要素  $f$ 、 $l$  を一度に算出する場合について示したが、行列要素  $f$ 、 $l$  をその成分ごとに計算してもよい。上記の式 3、式 4 から明らかなように、 $f_{12}$ 、 $l_{12}$  に固有ベクトル  $q_i$  を掛けたものが、行列要素  $f$ 、 $l$  の  $i$  番目の成分となる。このような方法で行列要素  $f$ 、 $l$  を計算する場合には、固有ベクトル固有ベクトル  $q_i$  のみを  $i$  の値を順次、変えながら記憶することによってその計算を行うことができる。したがって、行列要素  $f$ 、 $l$  の計算において行列  $Q$  全体を記憶する必要がなくなり、その計算で用いるメモリ上の作業領域を削減することが可能である。

20

【0106】

図 9 は、固有値を算出する処理について説明するための図である。固有値分解記憶部 16 では、固有値分解部 15 によって行列  $T_{111}$ 、 $T_{112}$ 、 $\dots$ 、 $T_{222}$  が固有値分解された結果、すなわち、それらの固有値と、行列要素とが記憶されているものとする。まず、固有値算出部 17 は、固有値等の 1 番目の算出を開始する (ステップ S401、S402)。ここで、固有値等の 1 番目の算出とは、図 9 の一番下の行の行列の固有値と、行列要素とから、下から 2 番目の行の行列の固有値と、行列要素とを算出することである。固有値算出部 17 は、行列  $T_{111}$  と、行列  $T_{112}$  との各固有値、及び行列  $T_{111}$  と、行列  $T_{112}$  との各行列要素を固有値分解記憶部 16 から読み出す。また、行列  $T_{111}$  を行列  $T_{111}$  と、行列  $T_{112}$  とに分解したときに発生した 2 個の値、を対角行列記憶部 13 から読み出す。そして、固有値算出部 17 は、それらの値を用いて、行列  $T_{111}$  の固有値を算出して固有値分解記憶部 16 に蓄積する (ステップ S501)。次に、固有値算出部 17 は、行列  $T_{111}$  の固有値を用いて直交行列  $Q$  を算出し (ステップ S502)、その直交行列  $Q$  を用いて行列  $T_{111}$  の行列要素を算出して固有値分解記憶部 16 に蓄積する (ステップ S503)。

30

40

【0107】

固有値等の 1 番目の算出はまだ終わっていないため (ステップ S404)、固有値算出部 17 は、上記説明と同様にして、行列  $T_{121}$ 、行列  $T_{122}$  の固有値、行列要素等に基づいて、行列  $T_{12}$  の固有値、行列要素を算出する (ステップ S402、S403)。このようにして、固有値等の 1 番目の算出が終了すると、固有値算出部 17 は、固有値等の 2 番目の算出、すなわち、行列  $T_{11}$ 、行列  $T_{12}$  の固有値、行列要素等に基づいて行列  $T_1$  の固有値、行列要素の算出を行う (ステップ S404、S405、S402、S403)。その後、固有値算出部 17 は、同様にして、行列  $T_{21}$ 、行列  $T_{22}$  の固有値、行列要素等に基づいて行列  $T_2$  の固有値、行列要素の算出を行う (ステップ S404、S

50

403)。

【0108】

固有値算出部17が固有値等の2番目の算出を終了すると(ステップS404)、次の固有値等の算出は、対称3重対角行列Tの固有値を求める最後の処理となるため(ステップS405, S402)、固有値算出部17は、固有値の算出のみを行い、その算出した固有値を固有値記憶部18に蓄積する(ステップS406)。このようにして、固有値の算出が終了する。

【0109】

なお、図9の行列 $T_1$ 、行列 $T_2$ の固有値、行列要素等に基づいて行列Tの固有値を算出する場合には、行列要素 $f_1$ 、 $l_2$ 、あるいは行列要素 $f_2$ 、 $l_1$ を用いるだけでよい。したがって、行列 $T_1$ 、行列 $T_2$ の行列要素を算出する場合に、行列要素 $f_1$ 、 $l_2$ 、あるいは行列要素 $f_2$ 、 $l_1$ を計算しなくてもよい。このようにすることで、計算量を削減することができ、処理スピードを向上させることが可能である。

10

【0110】

次に、固有ベクトル算出部19について説明する。本実施の形態では、固有ベクトル算出部19が(1)qd型ツイスト分解法により固有ベクトルを算出する場合と、(2)LV型ツイスト分解法により固有ベクトルを算出する場合とについて説明する。

【0111】

(1) qd型ツイスト分解法により固有ベクトルを算出する場合

図10は、qd型ツイスト分解法により固有ベクトルを算出する場合の固有ベクトル算出部19の構成を示す図である。図10において、固有ベクトル算出部19は、正定値化部21と、コレスキー分解部22と、ベクトル算出部23とを備える。

20

【0112】

正定値化部21は、固有値記憶部18から対称3重対角行列Tの固有値を読み出し、その固有値のいずれかが負の値であるかどうか判断する。そして、その固有値のいずれかが負の値である場合に、正定値化部21は、対角行列記憶部13から対称3重対角行列Tを読み出し、その対称3重対角行列Tの固有ベクトルを変化させることなく、対称3重対角行列Tのすべての固有値が正の値となるように、その対称3重対角行列Tを正定値化し、正定値化した後の固有値を固有値記憶部18に蓄積し、正定値化した後の対称3重対角行列Tを対角行列記憶部13に蓄積する。この正定値化の処理を行うのは、qd型ツイスト分解法により固有ベクトルを算出する場合には、対称3重対角行列Tの固有値がすべて正の値でなければならないからである。なお、対称3重対角行列Tを正定値化した行列も、対称3重対角行列Tと呼ぶことにする。また、この正定値化の処理が行われた場合には、コレスキー分解や固有ベクトルを算出する処理は、正定値化後の対称3重対角行列T、固有値に対して行われることになる。この処理の詳細については後述する。ここで、正定値化の行われた場合であっても、固有値記憶部18には、正定値化の前の固有値が記憶されているものとする。その正定値化の前の固有値が、固有値分解装置1が最終的に求める固有値だからである。また、正定値化の行われた場合であって、qd型ツイスト分解法による固有ベクトルの算出のみを行う場合には、対角行列記憶部13で記憶されている対称3重対角行列Tを、正定値化の後の対称3重対角行列Tで上書きしてもよい。一方、LV型ツイスト分解法による固有ベクトルの算出も行う場合には、対角行列記憶部13には、正定値化の前の対称3重対角行列Tと、正定値化の後の対称3重対角行列Tとが記憶されることになる。

30

40

【0113】

コレスキー分解部22は、対角行列記憶部13から、すべての固有値が正の値である対称3重対角行列Tを読み出し、固有値記憶部18から、対称3重対角行列Tの正の値の固有値を読み出す。そして、コレスキー分解部22は、対称3重対角行列Tの各要素に関してqd型変換を行うことによって、対称3重対角行列Tを上2重対角行列 $B^{(+1)}$ 及び下2重対角行列 $B^{(-1)}$ にコレスキー分解する。この処理の詳細については後述する。なお、「対称3重対角行列Tを上2重対角行列 $B^{(+1)}$ 及び下2重対角行列 $B^{(-1)}$

50

にコレスキー分解する」ことには、対称 3 重対角行列 T の固有値に単位ベクトルを掛けたものを対称 3 重対角行列 T から引いた行列を、上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  にコレスキー分解することが含まれてもよい。

【0114】

ベクトル算出部 23 は、コレスキー分解部 22 が算出した上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  の各要素と、対称 3 重対角行列 T の正の値の固有値とを用いて固有ベクトルを算出して固有ベクトル記憶部 20 に蓄積する。この固有ベクトルを各列に有する行列が、固有値分解の直交行列となる。

【0115】

次に、図 2 のフローチャートのステップ S103 の処理、すなわち、対称 3 重対角行列 T の固有ベクトルを算出する処理について、より詳細に説明する。まず、図 2 のフローチャートのステップ S103 の処理について、図 11 のフローチャートを用いて説明する。

10

【0116】

(ステップ S601) 正定値化部 21 は、固有値記憶部 18 から対称 3 重対角行列 T の固有値を読み出し、その固有値のいずれかが負の値であるかどうか判断する。そして、負の値がある場合には、ステップ S602 に進み、そうでない場合には、ステップ S603 に進む。

【0117】

(ステップ S602) 正定値化部 21 は、対角行列記憶部 13 から対称 3 重対角行列 T を読み出し、その対称 3 重対角行列 T の固有ベクトルを変化させることなく、対称 3 重対角行列 T のすべての固有値が正の値となるように、その対称 3 重対角行列 T を正定値化し、正定値化した後の対称 3 重対角行列 T を対角行列記憶部 13 に蓄積し、正定値化した後の固有値を固有値記憶部 18 に蓄積する。

20

【0118】

(ステップ S603) コレスキー分解部 22 は、対角行列記憶部 13 から、すべての固有値が正の値である対称 3 重対角行列 T を読み出し、固有値記憶部 18 から、対称 3 重対角行列 T の正の値の固有値を読み出す。なお、正定値化が行われた場合には、正定値化の後の対称 3 重対角行列 T、及び固有値が読み出されることになる。そして、コレスキー分解部 22 は、対称 3 重対角行列 T の各要素に関して qd 型変換を行うことによって、対称 3 重対角行列 T を上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  にコレスキー分解する。

30

【0119】

(ステップ S604) ベクトル算出部 23 は、コレスキー分解部 22 が算出した上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  の各要素と、対称 3 重対角行列 T の正の値の固有値とを用いて固有ベクトルを算出する。

【0120】

(ステップ S605) ベクトル算出部 23 は、算出した固有ベクトルを正規化する。すなわち、ベクトル算出部 23 は、算出した固有ベクトルのノルムを算出し、固有ベクトルを算出したノルムで割ったものを最終的な固有ベクトルとして固有ベクトル記憶部 20 に蓄積する。このようにして、対称 3 重対角行列 T を固有値分解する処理は終了となる。

40

なお、この後に、行列記憶部 11 が記憶している対称行列 A の固有ベクトルを算出する処理が行われてもよいが、ここでは省略している。

【0121】

また、固有ベクトル算出部 19 が算出した固有ベクトルの精度を上げるために、図 12 で示されるように、固有ベクトルに関する処理を実行してもよく、あるいは、実行しなくてもよい。すなわち、図示しない逆反復法処理部は、固有ベクトル記憶部 20 から固有ベクトル算出部 19 が算出した固有ベクトルを読み出し、その固有ベクトルに対して逆反復法の処理を実行し、その結果の固有ベクトルを固有ベクトル記憶部 20 に蓄積する (ステップ S701)。このように、逆反復法の処理を実行することによって、固有ベクトルの精度を向上させることができる。次に、図示しないグラムシュミット処理部は、高精度

50

が必要かどうか判断する（ステップ S 7 0 2）。この判断は、あらかじめ高精度が必要かどうか設定されている記録媒体等から、その設定を読み出すことによって判断してもよい。そして、図示しないグラムシュミット処理部は、高精度が必要な場合には、固有ベクトル記憶部 20 から逆反復法の処理の実行された固有ベクトルを読み出し、その固有ベクトルに対してグラムシュミット法の処理を実行し、その結果の固有ベクトルを固有ベクトル記憶部 20 に蓄積する（ステップ S 7 0 3）。このように、グラムシュミット法の処理を実行することによって、固有ベクトルの直交性を高めることができる。なお、逆反復法や、グラムシュミット法については、すでに公知であり、その詳細な説明を省略する。

【 0 1 2 2 】

[ 対称 3 重対角行列 T の正定値化 ]

行列 T を次のように固有値分解できたとする。

【 数 3 8 】

$$T = QDQ^T$$

【 0 1 2 3 】

その行列 T を次のように a 倍して s I だけシフトしたとする。ここで、I は行列 T と同じ大きさの単位行列であり、a, s は実数である。

【 数 3 9 】

$$T \rightarrow \hat{T} = aT + sI$$

【 0 1 2 4 】

すると、シフト後の固有値分解は、次のようになる。

【 数 4 0 】

$$\hat{T} = Q\hat{D}Q^T$$

ただし、 $\hat{D} = aD + sI$  である。

【 0 1 2 5 】

このように、行列 T を a 倍して s I だけシフトした場合には、各固有値が a 倍されて s だけシフトされるが、固有ベクトルは変化しない。したがって、この実数倍とシフトとによって、すべての固有値を正の値にしてから、ツイスト分解法を用いた固有ベクトルの算出を行えばよいことになる。

【 0 1 2 6 】

なお、算出された固有値に負の値が含まれる場合には、例えば、(A) 最小の固有値によりシフトを行ってもよく、(B) 最大の固有値によりシフトを行ってもよい。ここで、固有値には、負の値と正の値とが含まれているものとする。固有値がすべて正の値であれば、シフトを行う必要はなく、固有値がすべて負の値であれば、単に行列 T を - T とすれば、正定値化することができる。

【 0 1 2 7 】

(A) 最小の固有値によるシフト

最小の固有値を  $\lambda_{\min} < 0$  とすると、 $a = 1$ 、 $s = -\lambda_{\min} + \lambda_{\max} \times \epsilon$  とする。ここで、 $\epsilon$  としては、マシン・イプシロンに定数を掛けた値を用いてもよく、他の適切な値であってもよい。このように、行列 T をシフトすることによって、行列 T のすべての固有値を正の値にすることができる。

【 0 1 2 8 】

(B) 最大の固有値によるシフト

最大の固有値を  $\lambda_{\max} > 0$  とすると、 $a = -1$ 、 $s = \lambda_{\max} - \lambda_{\min} \times \epsilon$  とする。この場合には、次のように行列 T が変換されることになる。

【 数 4 1 】

$$T \rightarrow \hat{T} = -T + \lambda_{\max} - \lambda_{\min} \times \epsilon I = -(T - (\lambda_{\max} - \lambda_{\min} \times \epsilon)I)$$

10

20

30

40

50



## 【 0 1 2 9 】

上式の右辺の括弧内では、シフトによってすべての固有値が負の値になるが、全体にマイナスを掛けることによって、正定値化することができている。ここで、 $\lambda$  は、( A ) の場合と同様にマシン・イプシロンに定数 ( 例えば、2, 3, 4 などの実数であり、計算誤差を少なくするための値を選択することが好適である ) を掛けた値を用いてもよく、他の適切な値であってもよい。

## 【 0 1 3 0 】

固有値の最小値の絶対値が、固有値の最大値の絶対値よりも小さい場合には、上記 ( A ) のシフトを行い、固有値の最小値の絶対値が、固有値の最大値の絶対値よりも大きい場合には、上記 ( B ) のシフトを行うようにしてもよい。このようにすることで、シフトの量を少なくすることができ、丸め誤差を削減することが可能だからである。

なお、正定値化の後に固有ベクトルを算出する処理の詳細については、次の L V 型ツイスト分解法の場合と一緒に説明する。

## 【 0 1 3 1 】

( 2 ) L V 型ツイスト分解法により固有ベクトルを算出する場合

図 1 3 は、L V 型ツイスト分解法により固有ベクトルを算出する場合の固有ベクトル算出部 1 9 の構成を示す図である。図 1 3 において、固有ベクトル算出部 1 9 は、コレスキー分解部 3 1 と、ベクトル算出部 3 2 とを備える。

## 【 0 1 3 2 】

コレスキー分解部 3 1 は、対角行列記憶部 1 3 から、対称 3 重対角行列 T を読み出し、固有値記憶部 1 8 から、対称 3 重対角行列 T の固有値を読み出す。そして、コレスキー分解部 3 1 は、対称 3 重対角行列 T の各要素に関してミウラ変換、d L V v 型変換、逆ミウラ変換を行うことによって、対称 3 重対角行列 T を上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  にコレスキー分解する。この処理の詳細については後述する。なお、「対称 3 重対角行列 T を上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  にコレスキー分解する」ことには、対称 3 重対角行列 T の固有値に単位ベクトルを掛けたものを対称 3 重対角行列 T から引いた行列を、上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  にコレスキー分解することが含まれてもよい。

## 【 0 1 3 3 】

ベクトル算出部 3 2 は、コレスキー分解部 3 1 が算出した上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  の各要素と、対称 3 重対角行列 T の固有値とを用いて固有ベクトルを算出して固有ベクトル記憶部 2 0 に蓄積する。この固有ベクトルを各列に有する行列が、固有値分解の直交行列となる。

## 【 0 1 3 4 】

次に、図 2 のフローチャートのステップ S 1 0 3 の処理、すなわち、対称 3 重対角行列 T の固有ベクトルを算出する処理について、より詳細に説明する。まず、図 2 のフローチャートのステップ S 1 0 3 の処理について、図 1 4 のフローチャートを用いて説明する。

## 【 0 1 3 5 】

( ステップ S 8 0 1 ) コレスキー分解部 3 1 は、対角行列記憶部 1 3 から対称 3 重対角行列 T を読み出し、固有値記憶部 1 8 から対称 3 重対角行列 T の固有値を読み出す。そして、コレスキー分解部 3 1 は、対称 3 重対角行列 T の各要素に関してミウラ変換、d L V v 型変換、逆ミウラ変換を行うことによって、対称 3 重対角行列 T を上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  にコレスキー分解する。

## 【 0 1 3 6 】

( ステップ S 8 0 2 ) ベクトル算出部 3 2 は、上 2 重対角行列  $B^{(+1)}$  及び下 2 重対角行列  $B^{(-1)}$  の各要素と、対称 3 重対角行列 T の固有値とを用いて固有ベクトルを算出する。

## 【 0 1 3 7 】

( ステップ S 8 0 3 ) ベクトル算出部 3 2 は、算出した固有ベクトルを正規化する。すなわち、ベクトル算出部 3 2 は、算出した固有ベクトルのノルムを算出し、固有ベクトル

を算出したノルムで割ったものを最終的な固有ベクトルとして固有ベクトル記憶部 20 に蓄積する。このようにして、対称 3 重対角行列 T を固有値分解する処理は終了となる。

【 0 1 3 8 】

なお、この後に、行列記憶部 11 が記憶している対称行列 A の固有ベクトルを算出する処理が行われてもよいが、ここでは省略している。また、固有ベクトル算出部 19 が算出した固有ベクトルの精度を上げるために、図 12 で示されるように、固有ベクトルに関する処理を実行してもよく、あるいは、実行しなくてもよいことは、q d 型ツイスト分解法の場合と同様である。

【 0 1 3 9 】

[ 固有値からの固有ベクトルの算出 ]

固有値から固有ベクトルを算出する処理については、q d 型ツイスト分解法を用いる方法と、L V 型ツイスト分解法を用いる方法とについて説明する。図 11 のフローチャートで示されるように、q d 型ツイスト分解法を用いて固有ベクトルを算出する場合に、負の固有値があれば、対称 3 重対角行列 T を正定値化する処理をあらかじめ実行しておく（ステップ S 6 0 1, S 6 0 2）。q d 型ツイスト分解法を用いる場合には、この後の処理において、その正定値化の後の対称 3 重対角行列 T、及び固有値を用いるものとする。

10

【 0 1 4 0 】

対角行列記憶部 13 では、次のような  $m \times m$  の対称 3 重対角行列 T が記憶されているものとする。

【 数 4 2 】

$$T = \begin{pmatrix} t_1 & t_2 & & 0 \\ t_2 & t_3 & \ddots & \\ & \ddots & \ddots & t_{2m-2} \\ 0 & & t_{2m-2} & t_{2m-1} \end{pmatrix}$$

20

【 0 1 4 1 】

また、固有値記憶部 18 では、固有値  $\lambda_i$  ( $1 \leq i \leq m$ ) が記憶されているものとする。このような場合に、対称 3 重対角行列 T の固有ベクトルを求めることは、次の方程式の固有ベクトル  $x_j$  を求めることになる。

30

【 数 4 3 】

$$(T - \lambda_j I) x_j = \gamma_k e_k \quad (j=1,2,\dots,m)$$

ただし、行列 T は  $m \times m$  行列である。

$e_k$  は、 $k$  番目の要素が 1 で他の要素が 0 のベクトルである。

( $e_k$  は、単位行列 I の  $k$  番目の列である)

【 0 1 4 2 】

本来であれば、上記式の右辺は 0 になるはずであるが、行列 T の固有値を算出するときに、いくらかの誤差を含むため、固有ベクトル  $x_j$  が真値であれば、上記のように右辺に残差項が存在する。

40

【 0 1 4 3 】

ここで、以下のようにコレスキー分解できたとする。

## 【数 4 4】

$$T - \lambda_j I = \begin{cases} (B^{(+1)})^T B^{(+1)} \\ (B^{(-1)})^T B^{(-1)} \end{cases}$$

$$B^{(n)} \equiv \begin{pmatrix} b_1^{(n)} & b_2^{(n)} & & & \\ & b_3^{(n)} & \ddots & & \\ & & \ddots & b_{2m-2}^{(n)} & \\ \mathbf{0} & & & & b_{2m-1}^{(n)} \end{pmatrix}, \quad (n = 0, +1),$$

$$T \equiv B^{(0)T} B^{(0)}$$

$$B^{(-1)} \equiv \begin{pmatrix} b_1^{(-1)} & & & & \mathbf{0} \\ b_2^{(-1)} & b_3^{(-1)} & & & \\ & \ddots & \ddots & & \\ & & & b_{2m-2}^{(-1)} & b_{2m-1}^{(-1)} \end{pmatrix}$$

$$b_{2k-1}^{(n)} \equiv \xi_k^{(n)} \sqrt{q_k^{(n)}}, \quad b_{2k}^{(n)} \equiv \eta_k^{(n)} \sqrt{e_k^{(n)}},$$

$$(\xi_k^{(n)})^2 = 1, \quad (\eta_k^{(n)})^2 = 1, \quad (n = 0, +1, -1),$$

$$\xi_k^{(1)} \eta_k^{(1)} = \xi_k^{(0)} \eta_k^{(0)}, \quad \xi_{k+1}^{(-1)} \eta_k^{(-1)} = \xi_k^{(0)} \eta_k^{(0)}$$

10

20

## 【0 1 4 4】

ここで、行列 T の各要素と、行列 B<sup>(0)</sup> の各要素との対応は、次のようになる。

## 【数 4 5】

$$t_{2k-1} = (b_{2k-2}^{(0)})^2 + (b_{2k-1}^{(0)})^2 \quad k = 1, 2, \dots, m$$

$$t_{2k} = b_{2k-1}^{(0)} b_{2k}^{(0)} \quad k = 1, 2, \dots, m-1$$

ただし、 $b_0^{(0)} = 0$  である。

## 【0 1 4 5】

なお、行列 T の各要素と、行列 B<sup>(0)</sup> の各要素との対応を次のように書くこともできる。下記の式から明らかなように、行列 B<sup>(0)</sup> の各要素を、行列 T の各要素を用いて順次、算出することができる。

30

## 【数 4 6】

$$b_{2k-1}^{(0)} = \sqrt{t_{2k-1} - b_{2k-2}^{(0)2}} \quad k = 1, 2, \dots, m$$

$$b_{2k}^{(0)} = \frac{t_{2k}}{b_{2k-1}^{(0)}} \quad k = 1, 2, \dots, m-1$$

ただし、 $b_0^{(0)} = 0$  である。

## 【0 1 4 6】

上記のコレスキー分解した結果を、次式のように書くことができる。

40

【数 4 7】

$$T - \lambda_j I = \begin{cases} LD^+L^T \\ UD^-U^T \end{cases}$$

ただし、 $L, U, D$ は以下の通りである。

$$L \equiv \begin{pmatrix} 1 & & & \mathbf{0} \\ L_1 & 1 & & \\ & \ddots & \ddots & \\ & & L_{m-1} & 1 \end{pmatrix}, \quad L_k \equiv \frac{b_{2k}^{(+1)}}{b_{2k-1}^{(+1)}} \quad 10$$

$$U \equiv \begin{pmatrix} 1 & U_1 & & \\ & 1 & \ddots & \\ & & \ddots & U_{m-1} \\ \mathbf{0} & & & 1 \end{pmatrix}, \quad U_k \equiv \frac{b_{2k}^{(-1)}}{b_{2k+1}^{(-1)}}$$

$$D^\pm \equiv \text{diag}(D_1^\pm, D_2^\pm, \dots, D_m^\pm), \quad D_k^\pm \equiv (b_{2k-1}^{(\pm 1)})^2$$

【 0 1 4 7】

そして、次のようになる。

【数 4 8】

$$T - \lambda_j I = N_k D_k (N_k)^T \quad 20$$

$$N_k \equiv \begin{pmatrix} 1 & & & & & & \\ L_1 & 1 & & & & & \\ & \ddots & \ddots & & & & \\ & & L_{k-1} & 1 & U_k & & \\ & & & 1 & \ddots & & \\ & & & & \ddots & U_m & \\ & & & & & & 1 \end{pmatrix}, \quad 30$$

$$D_k \equiv \text{diag}(D_1^+, D_2^+, \dots, D_{k-1}^+, \gamma_k, D_{k+1}^-, \dots, D_m^-)$$

【 0 1 4 8】

ここで、行列  $N_k$  をツイスト行列と呼ぶ。また、

【数 4 9】

$$D_k \mathbf{e}_k = \gamma_k \mathbf{e}_k, \quad N_k \mathbf{e}_k = \mathbf{e}_k$$

であるので、 $(T - \lambda_j I) \mathbf{x}_j = \mathbf{e}_k$  は、

【数 5 0】

$$N_k^T \mathbf{x}_j = \mathbf{e}_k \quad 40$$

となる。この簡単な式を解くことによって固有ベクトルを算出することができる。具体的には、

【数 5 1】

$$x_j(\rho) = \begin{cases} 1, & \rho = k, \\ -L_\rho x_j(\rho+1), & \rho = k-1, k-2, \dots, 1, \\ -U_{\rho-1} x_j(\rho-1), & \rho = k+1, k+2, \dots, m \end{cases}$$

ただし、 $x_j(\rho)$ は $\mathbf{x}_j$ の $\rho$ 番目の要素である。

のようにわずかな演算で固有ベクトルを算出することができる。なお、ある  $0$  について、 $D_{0+} = 0$  あるいは  $D_{0-} = 0$  であったとしても、行列  $B^{(0)}$  の  $(\quad, \quad)$  成分である  $b_{2\quad-1}^{(0)}$  と、行列  $B$  の  $(\quad, \quad+1)$  成分である  $b_{2\quad}^{(0)}$  を用いて、  
あるいは、行列  $T$  の成分を用いて、

【数 5 2】

$$x_j(\rho_0) = \begin{cases} -\frac{b_{2\rho_0+1}^{(0)} b_{2\rho_0+2}^{(0)}}{b_{2\rho_0-1}^{(0)} b_{2\rho_0}^{(0)}} x_j(\rho_0+2), & \rho_0 < k, \\ -\frac{b_{2\rho_0-5}^{(0)} b_{2\rho_0-4}^{(0)}}{b_{2\rho_0-3}^{(0)} b_{2\rho_0-2}^{(0)}} x_j(\rho_0-2), & \rho_0 > k \end{cases}$$

$$= \begin{cases} -\frac{t_{2\rho_0+2}}{t_{2\rho_0}} x_j(\rho_0+2), & \rho_0 < k, \\ -\frac{t_{2\rho_0-4}}{t_{2\rho_0-2}} x_j(\rho_0-2), & \rho_0 > k \end{cases}$$

と固有ベクトルを算出することができる（このように固有ベクトルを算出する処理を例外処理と呼ぶことにする）。なお、残差項のパラメータ  $\lambda_k$  及び  $k$  の値は、

【数 5 3】

$$\gamma_k \equiv q_k^{(+1)} + q_k^{(-1)} - (e_{k-1}^{(0)} + q_k^{(0)} - \lambda_j) \quad (\text{式 5})$$

の絶対値が最小となるように決定する。したがって、上述のコレスキー分解を求め、ツイスト行列  $N_k$  を求めることができれば、固有ベクトルを求めることができることになる。そこで、次にコレスキー分解について説明する。

【0 1 4 9】

図 1 5 で示すように、 $T - j I$  をコレスキー分解することは、 $B^{(0)}$  に対応する  $\{q_k^{(0)}, e_k^{(0)}\}$  から、 $B^{(\pm 1)}$  に対応する  $\{q_k^{(\pm 1)}, e_k^{(\pm 1)}\}$  を求めることである。

【0 1 5 0】

(q d 型ツイスト分解法)

まず、q d 型ツイスト分解法で用いる q d 型変換について説明する（図 1 5 参照）。

【数 5 4】

$$q d \text{ 型変換} : \{q_k^{(0)}, e_k^{(0)}\} \mapsto \{q_k^{(\pm 1)}, e_k^{(\pm 1)}\}$$

この変換は、さらに stationary q d with shift (stqds) 変換

【数 5 5】

$$stqds \text{ 変換} : \{q_k^{(0)}, e_k^{(0)}\} \mapsto \{q_k^{(+1)}, e_k^{(+1)}\}$$

$$q_k^{(+1)} + e_{k-1}^{(+1)} = q_k^{(0)} + e_{k-1}^{(0)} - \lambda_j, \quad k = 1, 2, \dots, m,$$

$$q_k^{(+1)} e_k^{(+1)} = q_k^{(0)} e_k^{(0)}, \quad k = 1, 2, \dots, m-1,$$

$$e_0^{(0)} \equiv 0, \quad e_0^{(+1)} \equiv 0$$

10

20

30

40

50

及び reverse-time progressive qd with shift (rpqds) 変換

【数 5 6】

$$\begin{aligned} \text{rpqds 変換} : \{q_k^{(0)}, e_k^{(0)}\} &\mapsto \{q_k^{(-1)}, e_k^{(-1)}\} \\ q_k^{(-1)} + e_k^{(-1)} &= q_k^{(0)} + e_{k-1}^{(0)} - \lambda_j, \quad k=1,2,\dots,m, \\ q_{k+1}^{(-1)} e_k^{(-1)} &= q_k^{(0)} e_k^{(0)}, \quad k=1,2,\dots,m-1, \\ e_0^{(0)} &\equiv 0, \quad e_m^{(-1)} \equiv 0 \end{aligned}$$

に分けられる。固有値  $\lambda_j$  が既知であれば反復的な計算が不要なため、計算量を少なく抑えることができるが、常に数値安定性と精度が高い方法ではない。それは、stqds 変換、rpqds 変換は、共に減算による桁落ちが発生する可能性があるからである。例えば、stqds 変換において、 $q_k^{(0)} + e_{k-1}^{(0)} - \lambda_j \sim e_{k-1}^{(+1)}$  であれば、 $q_k^{(+1)}$  を求める際に、倍精度計算でもの有効数字がわずか 1 桁になることもある。その場合には、 $q_k^{(0)} e_k^{(0)} / q_k^{(+1)}$  を計算すると誤差が生じる。つまり、 $e_k^{(+1)}$  が精度よく計算できないことになる。また、 $q_{k+1}^{(+1)}$  を求めるのに  $e_k^{(+1)}$  が必要であり、 $e_k^{(+1)}$  を求めるのに  $q_k^{(+1)}$  が必要であるといったように逐次的な計算が要求されるので、1 箇所が発生した桁落ちによる誤差が波及し、さらなる誤差増大の可能性も秘めている。その結果、理論上は  $q_k^{(+1)} = 0$  であるが、誤差蓄積により  $q_k^{(+1)} = 0$  となり、 $q_k^{(0)} e_k^{(0)} / q_k^{(+1)}$  の計算においてオーバーフローが起こるといった数値不安定な状況も想定される。B<sup>(0)</sup> の成分  $\{b_{2k-1}^{(0)}, b_{2k}^{(0)}\}$  が与えられる、すなわち  $\{q_k^{(0)}, e_k^{(0)}\}$  が与えられると、 $\lambda_j$  及び  $e_{k-1}^{(+1)}$  が一意的に決まるので、この状況は避けることはできない。rpqds 変換も同様の性質を持つため、実用的なレベルにまで達したとはいいがたい。LAPACK において FORTRAN ルーチン DSTEGR として改良版が公開されているものの欠点は完全に解決されてはいない。

【0151】

なお、このように対角行列 T の各要素を、行列 B<sup>(0)</sup> の各要素に変換して、その行列 B<sup>(0)</sup> の各要素について qd 型変換を行う場合であっても、間接的に対角行列 T の各要素に関して qd 型変換を行うことになるため、「対角行列 T の各要素に関して qd 型変換を行う」と言うことにする。

また、ここではスタンダードな qd 型変換について説明したが、qd 型変換は differential qd 型変換であってもよい。

【0152】

(LV 型ツイスト分解法)

次に、LV 型ツイスト分解法で用いるミウラ変換、dLVv 型変換、逆ミウラ変換について説明する (図 15 参照)。

【数 5 7】

$$\begin{aligned} \text{ミウラ変換} : \{q_k^{(0)}, e_k^{(0)}\} &\mapsto \{u_{2k-1}^{(0)}, u_{2k}^{(0)}\} \\ \text{stdLVv 変換} : u_k^{(0)} &\mapsto u_k^{(+1)} \\ \text{rdLVv 変換} : u_k^{(0)} &\mapsto u_k^{(-1)} \\ \text{逆ミウラ変換} : \{u_{2k-1}^{(\pm 1)}, u_{2k}^{(\pm 1)}\} &\mapsto \{q_k^{(\pm 1)}, e_k^{(\pm 1)}\} \end{aligned}$$

【0153】

まず、ミウラ変換について説明する。この変換は、次のように示される。

10

20

30

40

【数 5 8】

$$u_{2k-1}^{(0)} = \frac{t_k^{(0)}}{\delta^{(0)}}, \quad k = 1, 2, \dots, m,$$

$$u_{2k}^{(0)} = \frac{e_k^{(0)}}{t_k^{(0)}}, \quad k = 1, 2, \dots, m-1,$$

$$t_k^{(0)} \equiv \frac{q_k^{(0)}}{\frac{1}{\delta^{(0)}} + u_{2k-2}^{(0)}} - 1,$$

$$u_0^{(0)} \equiv 0,$$

ただし、 $\delta^{(0)}$ は任意である。

10

【0 1 5 4】

次に、d L V v 型変換について説明する。この変換は、さらに stationary discrete Lotka - Volterra with variable step - size (std L V v) 変換

【数 5 9】

$$u_k^{(+1)} = u_k^{(0)} \times \frac{1 + \delta^{(0)} u_{k-1}^{(0)}}{1 + \delta^{(+1)} u_{k-1}^{(+1)}}, \quad k = 1, 2, \dots, 2m-1,$$

$$u_0^{(0)} \equiv 0, \quad u_0^{(+1)} \equiv 0$$

20

及び reverse - time discrete Lotka - Volterra with variable step - size (rd L V v) 変換

【数 6 0】

$$u_k^{(-1)} = u_k^{(0)} \times \frac{1 + \delta^{(0)} u_{k-1}^{(0)}}{1 + \delta^{(-1)} u_{k+1}^{(-1)}}, \quad k = 1, 2, \dots, 2m-1,$$

$$u_0^{(0)} \equiv 0, \quad u_{2m}^{(-1)} \equiv 0$$

に分けられる。ただし、

【数 6 1】

$$\frac{1}{\delta^{(0)}} - \frac{1}{\delta^{(n)}} = \lambda_j, \quad (n = +1, -1)$$

30

を満たす範囲内で  $(\pm 1)$  は任意である。

【0 1 5 5】

最後に、逆ミウラ変換について説明する。この変換は、次のように示される。

【数 6 2】

$$q_k^{(\pm 1)} = \frac{(1 + \delta^{(\pm 1)} u_{2k-2}^{(\pm 1)}) \times (1 + \delta^{(\pm 1)} u_{2k-1}^{(\pm 1)})}{\delta^{(\pm 1)}}, \quad k = 1, 2, \dots, m,$$

$$e_k^{(\pm 1)} = \delta^{(\pm 1)} u_{2k-1}^{(\pm 1)} u_{2k}^{(\pm 1)}, \quad k = 1, 2, \dots, m-1$$

$$u_0^{(\pm 1)} \equiv 0$$

40

【0 1 5 6】

このようにして、q d 型変換と同様に、コレスキー分解を行うことができる。q d 型変換では見られない離散 Lotka - Volterra 型変換の大きな特徴は、任意パラメータを持つことである。すなわち、 $\lambda_j = 1 / \delta^{(0)} - 1 / \delta^{(\pm 1)}$  を満たす範囲で  $\delta^{(n)}$  の値を任意に設定できる。 $\delta^{(n)}$  を変動させると補助変数  $u_k^{(n)}$  の値も変化するが、桁落ちによる誤差や数値不安定が発生するかどうかは事前に判定できる。この判定は、if 文によって実装されてもよい。この場合は、 $\delta^{(n)}$  を再設定後に再度計算すればよい。また、 $u_k^{(\pm 1)}$  が求めれば逆ミウラ変換によって  $q_k^{(\pm 1)}$  及び  $e_k$

50

( $\pm 1$ ) が独立に計算されるので、誤差が伝播しないという性質を持つ。なお、逆ミウラ変換をミウラ変換、ミウラ変換を逆ミウラ変換と呼んでもよく、stdLVv変換をstLVv変換と呼んでもよく、rdLVv変換をrLVv変換と呼んでもよい。

【0157】

なお、このように対角行列Tの各要素を、行列B<sup>(0)</sup>の各要素に変換して、その行列B<sup>(0)</sup>の各要素についてLV型変換を行う場合であっても、間接的に対角行列Tの各要素に関してLV型変換を行うことになるため、「対角行列Tの各要素に関してLV型変換を行う」と言うことにする。

【0158】

ここで、LV型ツイスト分解法による処理のより詳細な処理の一例について説明する。図16～図21は、LV型ツイスト分解法による処理の一例を示すフローチャートである。

10

【0159】

図16は、コレスキー分解の全体の処理の一例を示す図である。

(ステップS901)コレスキー分解部31は、ミウラ変換を行う。この処理の詳細については後述する。

【0160】

(ステップS902)コレスキー分解部31は、 $1 / (\pm 1)$  を  $1 / (0) - j$  とする。

(ステップS903)コレスキー分解部31は、後述するProcedure1の処理を実行する。

20

【0161】

(ステップS904)コレスキー分解部31は、後述するProcedure2の処理を実行する。

(ステップS905)コレスキー分解部31は、 $q_k^{(+1)}$  ,  $e_k^{(+1)}$  がすでに算出されているかどうか判断する。そして、すでに算出されている場合には、コレスキー分解の一連の処理は終了となり、一方、算出されていない場合には、ステップS901に戻る。

【0162】

図17は、図16のフローチャートにおけるステップS903の処理の詳細を示すフローチャートである。

30

(ステップS1001)コレスキー分解部31は、 $q_k^{(+1)}$  ,  $e_k^{(+1)}$  がすでに算出されているかどうか判断する。そして、すでに算出されている場合には、終了となり、算出されていない場合には、ステップS1002に進む。

(ステップS1002)コレスキー分解部31は、stdLVv変換等の処理を行う。この処理の詳細については後述する。

【0163】

図18は、図16のフローチャートにおけるステップS904の処理の詳細を示すフローチャートである。

(ステップS1101)コレスキー分解部31は、 $q_k^{(-1)}$  ,  $e_k^{(-1)}$  がすでに算出されているかどうか判断する。そして、すでに算出されている場合には、終了となり、算出されていない場合には、ステップS1102に進む。

40

(ステップS1102)コレスキー分解部31は、rdLVv変換等の処理を行う。この処理の詳細については後述する。

【0164】

図19は、図16のフローチャートのステップS901の処理の詳細を示すフローチャートである。

(ステップS1201)コレスキー分解部31は、 $1 / (0)$  の値を決定する。この値は、前述のように任意に決定することができるため、行列を正定値化する値に決定する。例えば、コレスキー分解部31は、 $1 / (0)$  の値を次式のように決定する。

50



## 【数 6 3】

$$-\frac{1}{\delta^{(0)}} = |\lambda_{\min}| + \varepsilon^{(0)} \times |\lambda_{\max}|$$

## 【0165】

なお、なお、 $\varepsilon^{(0)}$  は、マシン・イプシロンに定数（例えば、2, 3, 4 などの実数であり、計算誤差を少なくするための値を選択することが好適である）を掛けた値を用いてもよく、他の適切な値であってもよい。その後、例えば、ステップ S 1 2 0 3 等で桁落ちの発生する可能性があるとは判断された場合に、コレスキー分解部 3 1 は、 $1 / \varepsilon^{(0)}$  の値を決める  $\varepsilon^{(0)}$  の値を、例えば、マシン・イプシロンに掛ける定数を 1 ずつ大きくしながら設定していてもよい。なお、 $1 / \varepsilon^{(0)}$  の値を決定する方法は、これに限定されないことは言うまでもない。このように、L V 型ツイスト分解法では、この  $1 / \varepsilon^{(0)}$  の値を自由に設定することができるため、q d 型ツイスト分解法の場合のように、行列 T の正定値化を行わなくてもよいことになる。なお、q d 型ツイスト分解法の場合と同様に、行列 T の正定値化の処理を行ってから、L V 型ツイスト分解法を実行してもよいことは言うまでもない。

（ステップ S 1 2 0 2）コレスキー分解部 3 1 は、 $1$  を  $q_1 \varepsilon^{(0)} - 1 / \varepsilon^{(0)}$  に設定する。

## 【0166】

（ステップ S 1 2 0 3）コレスキー分解部 3 1 は、 $1$  の絶対値が  $1$  よりも大きいかどうか判断する。そして、大きい場合には、ステップ S 1 2 0 4 に進み、そうでない場合には、ステップ S 1 2 0 1 に戻って、 $1 / \varepsilon^{(0)}$  の値を決定する処理を再度実行する。なお、この  $\varepsilon^{(0)}$  も任意に定めることができる。例えば、 $\varepsilon^{(0)}$  としてマシン・イプシロンを用いてもよい。 $\varepsilon^{(0)}$  の値を大きくすると精度が向上し、 $\varepsilon^{(0)}$  の値を小さくすると精度が低下する。なお、このステップ S 1 2 0 3 で行っている処理は、桁落ちが発生する可能性について判断する処理である。 $1$  の絶対値が  $1$  よりも大きくない場合には、桁落ちの発生する可能性があるとは判断されることになる。

## 【0167】

（ステップ S 1 2 0 4）コレスキー分解部 3 1 は、 $2$  を  $q_1 \varepsilon^{(0)} / (1 + \varepsilon^{(0)} u_0 \varepsilon^{(0)})$  に設定する。なお、前述のように  $u_0 \varepsilon^{(0)} = 0$  であるため、 $2$  は  $q_1 \varepsilon^{(0)}$  に設定されたことになる。

## 【0168】

（ステップ S 1 2 0 5）コレスキー分解部 3 1 は、 $u_1 \varepsilon^{(0)} (1 + \varepsilon^{(0)} u_0 \varepsilon^{(0)})$  を  $1$  に設定する。ここで、 $1$  は、ステップ S 1 2 0 2 において  $q_1 \varepsilon^{(0)} - 1 / \varepsilon^{(0)}$  に設定されているが、前述のように  $u_0 \varepsilon^{(0)} = 0$  であるため、 $1$  は、 $q_1 \varepsilon^{(0)} / (1 + \varepsilon^{(0)} u_0 \varepsilon^{(0)}) - 1 / \varepsilon^{(0)}$  と等しく、これにミウラ変換を行うと  $u_1 \varepsilon^{(0)} = u_{2k-1} \varepsilon^{(0)} \mid_{k=1}$  となるからである。なお、 $u_0 \varepsilon^{(0)} = 0$  から  $(1 + \varepsilon^{(0)} u_0 \varepsilon^{(0)}) = 1$  である。

## 【0169】

（ステップ S 1 2 0 6）コレスキー分解部 3 1 は、カウンタ  $k$  を「1」に設定する。  
（ステップ S 1 2 0 7）コレスキー分解部 3 1 は、 $1$  を  $e_k \varepsilon^{(0)} / 1$  に設定する。前述のように、 $1$  は  $u_{2k-1} \varepsilon^{(0)}$  となるから、 $e_k \varepsilon^{(0)} / 1$  にミウラ変換を行うと、 $1$  は  $\varepsilon^{(0)} u_{2k} \varepsilon^{(0)}$  と等しいことになる。

## 【0170】

（ステップ S 1 2 0 8）コレスキー分解部 3 1 は、 $2$  を  $1 + 1$  に設定する。ステップ S 1 2 0 7 の説明からわかるように、 $2$  は  $1 + \varepsilon^{(0)} u_{2k} \varepsilon^{(0)}$  と等しいことになる。

## 【0171】

（ステップ S 1 2 0 9）コレスキー分解部 3 1 は、 $2$  の絶対値が  $1$  よりも大きいかどうか判断する。そして、大きい場合には、ステップ S 1 2 1 0 に進み、そうでない場合に

は、ステップ S 1 2 0 1 に戻って、 $1 / \left( \begin{matrix} 0 \\ \end{matrix} \right)$  の値を決定する処理を再度実行する。なお、このステップ S 1 2 0 9 で行っている処理は、ステップ S 1 2 0 3 の処理と同様に、桁落ちが発生する可能性について判断する処理である。 $2$  の絶対値が よりも大きくない場合には、桁落ちの発生する可能性があるとは判断されることになる。

【 0 1 7 2 】

(ステップ S 1 2 1 0) コレスキー分解部 3 1 は、 $1 \times 2$  を算出して  $u_{2k} \left( \begin{matrix} 0 \\ \end{matrix} \right)$   $(1 + \left( \begin{matrix} 0 \\ \end{matrix} \right) u_{2k-1} \left( \begin{matrix} 0 \\ \end{matrix} \right))$  に設定する。前述のように、 $1$  は  $\left( \begin{matrix} 0 \\ \end{matrix} \right) u_{2k} \left( \begin{matrix} 0 \\ \end{matrix} \right)$  と等しく、 $2$  は  $q_k \left( \begin{matrix} 0 \\ \end{matrix} \right) / (1 + \left( \begin{matrix} 0 \\ \end{matrix} \right) u_{2k-2} \left( \begin{matrix} 0 \\ \end{matrix} \right))$  であるため、 $1 \times 2$  にミウラ変換を実行すると、 $u_{2k} \left( \begin{matrix} 0 \\ \end{matrix} \right) (1 + \left( \begin{matrix} 0 \\ \end{matrix} \right) u_{2k-1} \left( \begin{matrix} 0 \\ \end{matrix} \right))$  に等しくなるからである。

10

【 0 1 7 3 】

(ステップ S 1 2 1 1) コレスキー分解部 3 1 は、 $2$  を  $q_{k+1} \left( \begin{matrix} 0 \\ \end{matrix} \right) / 2$  に設定する。前述のように、 $2$  は  $1 + \left( \begin{matrix} 0 \\ \end{matrix} \right) u_{2k} \left( \begin{matrix} 0 \\ \end{matrix} \right)$  と等しいため、 $2$  は、 $q_{k+1} \left( \begin{matrix} 0 \\ \end{matrix} \right) / (1 + \left( \begin{matrix} 0 \\ \end{matrix} \right) u_{2k} \left( \begin{matrix} 0 \\ \end{matrix} \right))$  となり、 $2$  における  $k$  の値を 1 だけインクリメントしたことになる。

【 0 1 7 4 】

(ステップ S 1 2 1 2) コレスキー分解部 3 1 は、 $1$  を  $2 - 1 / \left( \begin{matrix} 0 \\ \end{matrix} \right)$  に設定する。前述のように、 $2$  は  $q_{k+1} \left( \begin{matrix} 0 \\ \end{matrix} \right) / (1 + \left( \begin{matrix} 0 \\ \end{matrix} \right) u_{2k} \left( \begin{matrix} 0 \\ \end{matrix} \right))$  と等しいため、 $1$  は、 $q_{k+1} \left( \begin{matrix} 0 \\ \end{matrix} \right) / (1 + \left( \begin{matrix} 0 \\ \end{matrix} \right) u_{2k} \left( \begin{matrix} 0 \\ \end{matrix} \right)) - 1 / \left( \begin{matrix} 0 \\ \end{matrix} \right)$  となり、ミウラ変換を実行すると、 $1$  は、 $u_{2k+1} \left( \begin{matrix} 0 \\ \end{matrix} \right)$  となる。したがって、 $1$  における  $k$  の値を 1 だけインクリメントしたことになる。

20

【 0 1 7 5 】

(ステップ S 1 2 1 3) コレスキー分解部 3 1 は、 $1$  の絶対値が よりも大きいかどうか判断する。そして、大きい場合には、ステップ S 1 2 1 4 に進み、そうでない場合には、ステップ S 1 2 0 1 に戻って、 $1 / \left( \begin{matrix} 0 \\ \end{matrix} \right)$  の値を決定する処理を再度実行する。なお、このステップ S 1 2 1 3 で行っている処理は、ステップ S 1 2 0 3 の処理と同様に、桁落ちが発生する可能性について判断する処理である。 $1$  の絶対値が よりも大きくない場合には、桁落ちの発生する可能性があるとは判断されることになる。

【 0 1 7 6 】

(ステップ S 1 2 1 4) コレスキー分解部 3 1 は、 $2 \times 1$  を算出して  $u_{2k+1} \left( \begin{matrix} 0 \\ \end{matrix} \right) (1 + \left( \begin{matrix} 0 \\ \end{matrix} \right) u_{2k} \left( \begin{matrix} 0 \\ \end{matrix} \right))$  に設定する。前述のように、 $2$  は  $1 + \left( \begin{matrix} 0 \\ \end{matrix} \right) u_{2k} \left( \begin{matrix} 0 \\ \end{matrix} \right)$  と等しく、 $1$  は、 $u_{2k+1} \left( \begin{matrix} 0 \\ \end{matrix} \right)$  に等しいからである。

30

(ステップ S 1 2 1 5) コレスキー分解部 3 1 は、カウンタ  $k$  を 1 だけインクリメントする。

【 0 1 7 7 】

(ステップ S 1 2 1 6) コレスキー分解部 3 1 は、カウンタ  $k$  が  $m$  に等しいかどうか判断する。そして、 $m$  に等しい場合には、一連の処理は終了となり、そうでない場合には、ステップ S 1 2 0 7 に戻る。

【 0 1 7 8 】

このようにして、 $u_{2k+1} \left( \begin{matrix} 0 \\ \end{matrix} \right) (1 + \left( \begin{matrix} 0 \\ \end{matrix} \right) u_{2k} \left( \begin{matrix} 0 \\ \end{matrix} \right))$  と、 $u_{2k} \left( \begin{matrix} 0 \\ \end{matrix} \right) (1 + \left( \begin{matrix} 0 \\ \end{matrix} \right) u_{2k-1} \left( \begin{matrix} 0 \\ \end{matrix} \right))$  とが算出されることになる。これらの値は、コレスキー分解部 3 1 が有する図示しないメモリ等において一時的に記憶されてもよい。

40

【 0 1 7 9 】

図 20 は、図 17 のフローチャートのステップ S 1 0 0 2 の処理の詳細を示すフローチャートである。

(ステップ S 1 3 0 1) コレスキー分解部 3 1 は、 $2$  を  $1 + \left( \begin{matrix} + 1 \\ \end{matrix} \right) u_0 \left( \begin{matrix} + 1 \\ \end{matrix} \right)$  に設定する。なお、前述のように  $u_0 \left( \begin{matrix} + 1 \\ \end{matrix} \right) = 0$  であるため、 $2$  は 1 に設定されたことになる。

【 0 1 8 0 】

(ステップ S 1 3 0 2) コレスキー分解部 3 1 は、 $1$  を  $u_1 \left( \begin{matrix} 0 \\ \end{matrix} \right) (1 + \left( \begin{matrix} 0 \\ \end{matrix} \right) u$

50

$u_0^{(0)}) / (1 + u_0^{(+1)})$  に設定する。ここで、 $u_1^{(0)} (1 + u_0^{(0)})$  の値としては、ステップ S 1 2 0 5 で算出したものを用いる。なお、前述のように  $u_0^{(+1)} = 0$  である。また、この 1 の式に s t d L V v 変換を実行すると、1 は、 $u_1^{(+1)}$  に設定されたことになる。

【0181】

(ステップ S 1 3 0 3) コレスキー分解部 3 1 は、カウンタ k を「1」に設定する。

(ステップ S 1 3 0 4) コレスキー分解部 3 1 は、1 を  $1 + 1 / u_0^{(+1)}$  に設定する。

【0182】

(ステップ S 1 3 0 5) コレスキー分解部 3 1 は、1 の絶対値が よりも大きいかどうか判断する。そして、大きい場合には、ステップ S 1 3 0 6 に進み、そうでない場合には、図 1 6 のフローチャートの Procedure 2 (ステップ S 9 0 4) に進む。なお、このステップ S 1 3 0 5 で行っている処理は、ステップ S 1 2 0 3 の処理と同様に、桁落ちが発生する可能性について判断する処理である。1 の絶対値が よりも大きくない場合には、桁落ちの発生する可能性があるとして判断されることになる。

10

【0183】

(ステップ S 1 3 0 6) コレスキー分解部 3 1 は、2 を  $u_{2k}^{(0)} (1 + u_{2k-1}^{(0)}) / 1$  に設定する。ここで、 $u_{2k}^{(0)} (1 + u_{2k-1}^{(0)})$  の値としては、ステップ S 1 2 1 0 で算出したものを用いる。また、この 2 の式に s t d L V v 変換を実行すると、2 は、 $(+1) u_{2k}^{(+1)}$  に設定されたことになる。

20

【0184】

(ステップ S 1 3 0 7) コレスキー分解部 3 1 は、 $1 \times 2$  を算出する。前述のように、1 は  $1 + 1 / u_0^{(+1)} = u_{2k-1}^{(+1)} + 1 / u_0^{(+1)}$  に等しく、2 は  $1 + (+1) u_{2k-2}^{(+1)}$  に等しいため、 $1 \times 2$  に逆ミウラ変換を実行すると  $q_k^{(+1)}$  に等しくなる。したがって、コレスキー分解部 3 1 は、 $1 \times 2$  を算出して  $q_k^{(+1)}$  に設定する。

【0185】

(ステップ S 1 3 0 8) コレスキー分解部 3 1 は、2 を  $1 + 2$  に設定する。前述のように、2 は  $(+1) u_{2k}^{(+1)}$  と等しいため、2 は  $1 + (+1) u_{2k}^{(+1)}$  となる。したがって、2 における k の値を 1 だけインクリメントしたことになる。

30

【0186】

(ステップ S 1 3 0 9) コレスキー分解部 3 1 は、2 の絶対値が よりも大きいかどうか判断する。そして、大きい場合には、ステップ S 1 3 1 0 に進み、そうでない場合には、図 1 6 のフローチャートの Procedure 2 (ステップ S 9 0 4) に進む。なお、このステップ S 1 3 0 9 で行っている処理は、ステップ S 1 2 0 3 の処理と同様に、桁落ちが発生する可能性について判断する処理である。2 の絶対値が よりも大きくない場合には、桁落ちの発生する可能性があるとして判断されることになる。

【0187】

(ステップ S 1 3 1 0) コレスキー分解部 3 1 は、 $1 \times 2$  を算出する。前述のように、1 は  $u_{2k-1}^{(+1)}$  に等しく、2 は  $(+1) u_{2k}^{(+1)}$  と等しいため、 $1 \times 2$  に逆ミウラ変換を実行すると、 $e_k^{(+1)}$  に等しくなる。したがって、コレスキー分解部 3 1 は、 $1 \times 2$  を算出して  $e_k^{(+1)}$  に設定する。

40

【0188】

(ステップ S 1 3 1 1) コレスキー分解部 3 1 は、1 を  $u_{2k+1}^{(0)} (1 + u_{2k}^{(0)}) / 2$  に設定する。ここで、 $u_{2k+1}^{(0)} (1 + u_{2k}^{(0)})$  の値としては、ステップ S 1 2 1 4 で算出したものを用いる。また、この 1 の式に s t d L V v 変換を実行すると、1 は、 $u_{2k+1}^{(+1)}$  に設定されたことになる。したがって、1 における k の値を 1 だけインクリメントしたことになる。

50

## 【0189】

(ステップS1312) コレスキー分解部31は、カウンタkを1だけインクリメントする。

(ステップS1313) コレスキー分解部31は、カウンタkがmに等しいかどうか判断する。そして、mに等しい場合には、ステップS1314に進み、そうでない場合には、ステップS1304に戻る。

## 【0190】

(ステップS1314) コレスキー分解部31は、 $2 \times (1 + 1 / (\dots))$ を算出する。これは、ステップS1304で1を更新した後に、ステップS1307で $1 \times 2$ を計算することと等しい。したがって、コレスキー分解部31は、 $2 \times (1 + 1 / (\dots))$ を算出して $q_1 (\dots)$ に設定する。このようにして、ミウラ変換された結果にstdLVv変換と、逆ミウラ変換とを実行して、 $q_k (\dots)$ 、 $e_k (\dots)$ を算出する処理が終了する。これらの値は、コレスキー分解部31が有する図示しないメモリ等において一時的に記憶されてもよい。

10

## 【0191】

図21は、図18のフローチャートのステップS1102の処理の詳細を示すフローチャートである。

(ステップS1401) コレスキー分解部31は、1を $u_{2m-1}^{(0)} (1 + \dots)$ に設定する。ここで、 $u_{2m-1}^{(0)} (1 + \dots)$ の値としては、ステップS1214で算出したものを用いる。なお、前述のように $u_{2m}^{(-1)} = 0$ である。また、この1の式にrdLVv変換を実行すると、1は、 $u_{2m-1}^{(-1)}$ に設定されたことになる。

20

## 【0192】

(ステップS1402) コレスキー分解部31は、カウンタkを「m-1」に設定する。

(ステップS1403) コレスキー分解部31は、1を $1 + 1 / (\dots)$ に設定する。

## 【0193】

(ステップS1404) コレスキー分解部31は、1の絶対値がよりも大きいかどうか判断する。そして、大きい場合には、ステップS1405に進み、そうでない場合には、図16のフローチャートのミウラ変換(ステップS901)に戻る。なお、このステップS1404で行っている処理は、ステップS1203の処理と同様に、桁落ちが発生する可能性について判断する処理である。1の絶対値がよりも大きくない場合には、桁落ちの発生する可能性があるとして判断されることになる。

30

## 【0194】

(ステップS1405) コレスキー分解部31は、2を $u_{2k}^{(0)} (1 + \dots)$ に設定する。ここで、 $u_{2k}^{(0)} (1 + \dots)$ の値としては、ステップS1210で算出したものを用いる。また、この2の式にrdLVv変換を実行すると、2は、 $u_{2k}^{(-1)}$ に設定されたことになる。

40

## 【0195】

(ステップS1406) コレスキー分解部31は、2を $1 + 2$ に設定する。前述のように、2は $u_{2k}^{(-1)}$ と等しいため、2は $1 + u_{2k}^{(-1)}$ となる。

## 【0196】

(ステップS1407) コレスキー分解部31は、2の絶対値がよりも大きいかどうか判断する。そして、大きい場合には、ステップS1408に進み、そうでない場合には、図16のフローチャートのミウラ変換(ステップS901)に戻る。なお、このステップS1407で行っている処理は、ステップS1203の処理と同様に、桁落ちが発生

50

する可能性について判断する処理である。2の絶対値が1よりも大きくない場合には、桁落ちの発生する可能性があることが判断されることになる。

【0197】

(ステップS1408) コレスキー分解部31は、 $1 \times 2$ を算出する。前述のように、 $1$ は  $1 + 1 / (\cdot^{-1}) = u_{2k+1}(\cdot^{-1}) + 1 / (\cdot^{-1})$  に等しく、 $2$ は  $1 + (\cdot^{-1}) u_{2k}(\cdot^{-1})$  に等しいため、 $1 \times 2$ に逆ミウラ変換を実行すると  $q_{k+1}(\cdot^{-1})$  に等しくなる。したがって、コレスキー分解部31は、 $1 \times 2$ を算出して  $q_{k+1}(\cdot^{-1})$  に設定する。

【0198】

(ステップS1409) コレスキー分解部31は、 $1$ を  $u_{2k-1}(\cdot^0)(1 + (\cdot^0) u_{2k-2}(\cdot^0)) / 2$  に設定する。ここで、 $u_{2k-1}(\cdot^0)(1 + (\cdot^0) u_{2k-2}(\cdot^0))$ の値としては、ステップS1205, S1214で算出したものを用いる。また、この $1$ の式に  $s t d L V v$ 変換を実行すると、 $1$ は、 $u_{2k-1}(\cdot^{-1})$  に設定されたことになる。したがって、 $1$ における $k$ の値を1だけデクリメントしたことになる。

10

【0199】

(ステップS1410) コレスキー分解部31は、 $1 \times 2$ を算出する。前述のように、 $1$ は  $u_{2k-1}(\cdot^{-1})$  に等しく、 $2$ は  $(\cdot^{-1}) u_{2k}(\cdot^{-1})$  と等しいため、 $1 \times 2$ に逆ミウラ変換を実行すると、 $e_k(\cdot^{-1})$  に等しくなる。したがって、コレスキー分解部31は、 $1 \times 2$ を算出して  $e_k(\cdot^{-1})$  に設定する。

20

【0200】

(ステップS1411) コレスキー分解部31は、カウンタ $k$ を1だけデクリメントする。

(ステップS1412) コレスキー分解部31は、カウンタ $k$ が0に等しいかどうか判断する。そして、0に等しい場合には、ステップS1413に進み、そうでない場合には、ステップS1403に戻る。

【0201】

(ステップS1413) コレスキー分解部31は、 $1 + 1 / (\cdot^{-1})$ を算出する。これは、ステップS1403で $1$ を更新した後に、ステップS1408で $1 \times 2$ を計算することと等しい。この場合、 $2$ は1だからである。したがって、コレスキー分解部31は、 $1 + 1 / (\cdot^{-1})$ を算出して  $q_1(\cdot^{-1})$  に設定する。このようにして、ミウラ変換された結果に  $r d L V v$ 変換と、逆ミウラ変換とを実行して、 $q_k(\cdot^{-1})$ 、 $e_k(\cdot^{-1})$ を算出する処理が終了する。これらの値は、コレスキー分解部31が有する図示しないメモリ等において一時的に記憶されてもよい。

30

【0202】

なお、図16のフローチャートの処理によって算出することができるのは1個の固有値に対応する固有ベクトルであるため、全ての固有値に対応する固有ベクトルを算出する場合には、コレスキー分解部31は、図16のフローチャートの処理を固有値の数だけ繰り返すことになる。

【0203】

メモリ消費を抑えるために、補助変数  $u_k(\cdot^n)$  のための配列は必ずしも用意する必要はない。一方、 $1 + (\cdot^0) u(\cdot^0)$  のためのメモリ領域を確保し、ミウラ変換、 $d L V v$ 型変換、逆ミウラ変換のステップにまたがってこの値を利用することで、メモリ消費を抑え、計算量を低減することができる。これにより、誤差も低減される。

40

【0204】

ここで、誤差について説明する。人間が理想的な状況、すなわち、無限桁の計算をいくらでもできるとすると、 $q d$ 型ツイスト分解法であっても、 $L V$ 型ツイスト分解法であっても問題ない。しかしながら、コンピュータで計算を行う場合は注意が必要である。有限桁の計算しか行えないコンピュータ上では、数学的に正しい計算法を使っても必ずしも正しい結果が得られる訳ではない。そればかりか、いつまでたっても計算が終了しな

50

いといった思わぬ数値的な問題が発生する場合もある。

【0205】

コンピュータ計算による誤差には、丸め誤差及び桁落ちによる誤差などが知られている。丸め誤差単独では、高々有効桁の最後の桁が真値と比べて異なる程度で大きな誤差にはならない。また、指数部が異なる2つの実数の加算、乗算、除算の少なくとも1つの演算を行えばやはり丸め誤差が生じるが、それ以上の誤差は発生しない。さらに、このような丸め誤差を発生するような操作が繰り返されても、丸めモードがnear(四捨五入)ならば、一方的に切り上げられたり、あるいは切り捨てられたりして誤差が極端に蓄積することは少ない。よって、多くの数値計算法は加算、乗算、除算の少なくとも1つの演算によって発生する丸め誤差を特別注意することは少なく、適切な固有値計算でも結果的に丸め誤差は一様に増大しない。

10

【0206】

問題となるのは、同符号の実数の減算及び異符号の実数の加算により生じる、桁落ちである。桁落ちによる誤差で値が0となった後、その値による除算を行うと、0が分母にくるような不定形となり計算不可能となる。こうなるといつまでたっても計算が終了しない。減算 除算と計算する部分がqd型ツイスト分解法、LV型ツイスト分解法の両方に存在するので、桁落ち誤差には十分に注意する必要がある。

【0207】

LV型ツイスト分解法では、上述の桁落ちによる誤差を含んでいるかどうかは減算によって得られた値が小さいかどうかで判断できる。qd型ツイスト分解法の場合、桁落ち誤差を含むことが分かったとしても、それを回避することはできない。なぜならば、初期値として $\{q_k^{(0)}, e_k^{(0)}\}$ が与えられると、 $j$ は一意的に決定され、 $\{q_k^{(\pm 1)}, e_k^{(\pm 1)}\}$ も一意的に導出されるためである。すなわち、任意パラメータを持たない自由度のない計算法であるためである。

20

【0208】

それに対して、LV型ツイスト分解法は、自由に設定できるパラメータ $(0)$ を持つため、補助変数 $u_k^{(n)}$ の値を様々に変化させることができる(図22参照)。すなわち、様々な経路で $\{q_k^{(\pm 1)}, e_k^{(\pm 1)}\}$ を計算することができる。よって、桁落ちが発生する場合も回避できる。図19~図21の条件判定によって桁落ちの影響をチェックし、減算によって得られた値の絶対値が小さな数より大きいという条件が満たされなければ、パラメータ $(0)$ の設定に戻るといものである。この処理は、条件が満たされるまで繰り返される。なお、精度よりも高速性を重視する場合は、数回条件が満たされなければ $(q_k^{(+1)} = 0$ あるいは $q_k^{(-1)} = 0$ ならば)、例外処理を行ってもよい。

30

【0209】

なお、前述の行列Tの各要素と、行列 $B^{(0)}$ の各要素との対応を示す式を用いると、qd型変換を次のように書き換えることもできる。

【数64】

qd型変換： $\{t_{2k-1}, t_{2k}\} \mapsto \{q_k^{(\pm 1)}, e_k^{(\pm 1)}\}$

40

【0210】

この変換は、さらにstqds変換

【数65】

stqds変換： $\{t_{2k-1}, t_{2k}\} \mapsto \{q_k^{(+1)}, e_k^{(+1)}\}$

$$q_k^{(+1)} + e_{k-1}^{(+1)} = t_{2k-1} - \lambda_j, \quad k = 1, 2, \dots, m,$$

$$q_k^{(+1)} e_k^{(+1)} = (t_{2k})^2, \quad k = 1, 2, \dots, m-1,$$

$$e_0^{(+1)} \equiv 0$$

及びrpqds変換

50

## 【数 6 6】

r p q d s 変換:  $\{t_{2k-1}, t_{2k}\} \mapsto \{q_k^{(-1)}, e_k^{(-1)}\}$

$$q_k^{(-1)} + e_k^{(-1)} = t_{2k-1} - \lambda_j, \quad k = 1, 2, \dots, m,$$

$$q_{k+1}^{(-1)} e_k^{(-1)} = (t_{2k})^2, \quad k = 1, 2, \dots, m-1,$$

$$e_m^{(-1)} \equiv 0$$

に分けられる。このように、 $\{t_{2k-1}, t_{2k}\}$  から  $\{q_k^{(\pm 1)}, e_k^{(\pm 1)}\}$  を求める場合には、 $\{t_{2k-1}, t_{2k}\}$  から  $\{q_k^{(0)}, e_k^{(0)}\}$  を求め、さらに  $\{q_k^{(0)}, e_k^{(0)}\}$  から  $\{q_k^{(\pm 1)}, e_k^{(\pm 1)}\}$  を求める場合に比べて、計算量が少なくなることとなり、計算速度を向上させることも可能である。

10

## 【0 2 1 1】

なお、この場合には、対角行列 T の各要素について直接 q d 型変換が行われることになる。したがって、この変換が、「対角行列 T の各要素に関して q d 型変換を行う」場合に含まれることは言うまでもない。

## 【0 2 1 2】

また、q d 型変換と同様に、前述の行列 T の各要素と、行列 B<sup>(0)</sup> の各要素との対応を示す式を用いると、L V 型変換におけるミウラ変換を次のように書き換えることもできる。

## 【数 6 7】

ミウラ変換:  $\{t_{2k-1}, t_{2k}\} \mapsto \{u_{2k-1}^{(0)}, u_{2k}^{(0)}\}$

s t d L V v 変換:  $u_k^{(0)} \mapsto u_k^{(+1)}$

r d L V v 変換:  $u_k^{(0)} \mapsto u_k^{(-1)}$

逆ミウラ変換:  $\{u_{2k-1}^{(\pm 1)}, u_{2k}^{(\pm 1)}\} \mapsto \{q_k^{(\pm 1)}, e_k^{(\pm 1)}\}$

20

## 【0 2 1 3】

この場合のミウラ変換は、次のように示される。

## 【数 6 8】

$$t_{2k-1} = \frac{1}{\delta^{(0)}} (1 + \delta^{(0)} u_{2k-2}^{(0)}) (1 + \delta^{(0)} u_{2k-1}^{(0)}) + \delta^{(0)} u_{2k-3}^{(0)} u_{2k-2}^{(0)} \quad k = 1, 2, \dots, m$$

30

$$(t_{2k})^2 = (1 + \delta^{(0)} u_{2k-2}^{(0)}) (1 + \delta^{(0)} u_{2k-1}^{(0)}) u_{2k-1}^{(0)} u_{2k}^{(0)} \quad k = 1, 2, \dots, m-1$$

$$u_0^{(0)} \equiv 0,$$

ただし、 $\delta^{(0)}$  は任意である。

このように、 $\{t_{2k-1}, t_{2k}\}$  から  $\{u_{2k-1}^{(0)}, u_{2k}^{(0)}\}$  を求める場合には、 $\{t_{2k-1}, t_{2k}\}$  から  $\{q_k^{(0)}, e_k^{(0)}\}$  を求め、さらに  $\{q_k^{(0)}, e_k^{(0)}\}$  から  $\{u_{2k-1}^{(0)}, u_{2k}^{(0)}\}$  を求める場合に比べて、計算量が少なくなることとなり、計算速度を向上させることも可能である。

40

## 【0 2 1 4】

なお、この場合には、対角行列 T の各要素について直接 L V 型変換が行われることになる。したがって、この変換が、「対角行列 T の各要素に関して L V 型変換を行う」場合に含まれることは言うまでもない。

## 【0 2 1 5】

q d 型ツイスト分解法あるいは L V 型ツイスト分解法によってコレスキー分解をすることができると、上述のようにツイストされた行列  $N_k$  を算出することができ、その行列  $N_k$  の  $N_k^T x_j = e_k$  を解くことによって、 $x_j$  を算出することができる。ここで、

【数 6 9】

$$\mathbf{x}_j \leftarrow \frac{\mathbf{x}_j}{\|\mathbf{x}_j\|}$$

と  $\mathbf{x}_j$  を置き換えることにより、この  $\mathbf{x}_j$  を正規化する。このようにして、固有ベクトル  $\mathbf{x}_j$  を求めることができる。この固有ベクトル  $\mathbf{x}_j$  を用いて、 $Q_T = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m)$  とすることにより、直交行列  $Q_T$  を算出することができる。

【0 2 1 6】

したがって、次式のように対称 3 重対角行列  $T$  について固有値分解が行われたことになる。

【数 7 0】

$$T = Q_T D_T Q_T^T$$

【0 2 1 7】

また、対称 3 重対角行列  $T$  の固有値分解を行うことができれば、次のように、対称行列  $A$  の固有値分解も行われる。

【数 7 1】

$$\begin{aligned} A &= PTP^T \\ &= P(Q_T D_T Q_T^T)P^T \\ &= (PQ_T)D_T(PQ_T)^T \\ &= Q_A D_T Q_A^T \end{aligned}$$

【0 2 1 8】

固有ベクトル算出部 19 は、上述のようにして、固有値記憶部 18 が記憶している対称 3 重対角行列  $T$  の固有値と、対角行列記憶部 13 が記憶している対称 3 重対角行列  $T$  とを用いて、対称 3 重対角行列  $T$  の固有ベクトルを算出する。まず、コレスキー分解部 31 は、対角行列記憶部 13 から対称 3 重対角行列  $T$  を読み出し、固有値記憶部 18 から対称 3 重対角行列  $T$  の固有値を読み出す。そして、コレスキー分解部 31 は、図 23 のフローチャートで示されるように各変換を行い、コレスキー分解の処理を行う。ここでは、 $L V$  型ツイスト分解法を用いる場合について説明する。

【0 2 1 9】

コレスキー分解部 31 は、まず、対称 3 重対角行列  $T$  の各要素の値から、 $q_k^{(0)}$ 、 $e_k^{(0)}$  を求める。そして、コレスキー分解部 31 は、ミウラ変換を実行することにより、 $u_1^{(0)}$ 、 $u_2^{(0)}$  等を順次求めていく（ステップ S1501）。

【0 2 2 0】

次に、コレスキー分解部 31 は、ミウラ変換で得られた  $u_k^{(0)}$  を用いて、 $stdLV$  変換を実行することにより、 $u_1^{(+1)}$  等を順次求めていく（ステップ S1502）。また、コレスキー分解部 31 は、ミウラ変換で得られた  $u_k^{(0)}$  を用いて、 $rdLV$  変換を実行することにより、 $u_{2m-1}^{(-1)}$  等を順次求めていく（ステップ S1503）。

【0 2 2 1】

最後に、コレスキー分解部 31 は、 $stdLV$  変換で得られた  $u_k^{(+1)}$  及び  $rdLV$  変換で得られた  $u_k^{(-1)}$  を用いて、逆ミウラ変換を実行することにより、 $q_1^{(\pm 1)}$ 、 $e_1^{(\pm 1)}$  等を順次求めていく（ステップ S1504）。 $q_k^{(\pm 1)}$  及び  $e_k^{(\pm 1)}$  が求まると、すなわち、上 2 重対角行列  $B^{(+1)}$  と、下 2 重対角行列  $B^{(-1)}$  が求まると、コレスキー分解部 31 は、それらをベクトル算出部 32 に渡す。なお、コレスキー分解部 31 は、各固有値について、それぞれコレスキー分解を行うものとする。

【0 2 2 2】

10

20

30

40

50



なお、ここでは、説明の便宜上、図 2 3 のフローチャートを用いてコレスキー分解の処理を説明したが、図 1 6 ~ 図 2 1 のフローチャートで示されるように処理を行ってもよいことは言うまでもない。

【 0 2 2 3 】

ベクトル算出部 3 2 は、固有値記憶部 1 8 から固有値を読み出し、式 5 を用いて  $k$  の値を決定する。ベクトル算出部 3 2 は、その  $k$  の値を用いて、 $q_k^{(\pm 1)}$  及び  $e_k^{(\pm 1)}$  からツイスト行列  $N_k$  を求め、 $N_k^T x_j = e_k$  を解くことによって、 $x_j$  を算出する (ステップ S 8 0 2)。

【 0 2 2 4 】

次に、ベクトル算出部 3 2 は、算出した  $x_j$  を正規化して、固有ベクトル  $x_j$  を求めて固有ベクトル記憶部 2 0 に蓄積する (ステップ S 8 0 3)。なお、ベクトル算出部 3 2 は、各固有値について、 $x_j$  を算出する処理と、正規化する処理とを行うことによって、全ての固有ベクトルを算出することができる。このようにして、固有ベクトルの算出が終了する。

この後、行列記憶部 1 1 が記憶している対称行列  $A$  の固有ベクトルを算出してもよい。その算出の方法は、前述の通りである。

【 0 2 2 5 】

また、固有値算出部 1 7 が算出した固有値、固有ベクトル算出部 1 9 が算出した固有ベクトルを出力する図示しない出力部を固有値分解装置 1 が備えてもよい。ここで、図示しない出力部による出力は、例えば、固有値等の表示デバイス (例えば、CRT や液晶ディスプレイなど) への表示でもよく、固有値等の所定の機器への通信回線を介した送信でもよく、固有値等のプリンタによる印刷でもよく、固有値等の記録媒体への蓄積でもよい。なお、その出力部は、出力を行うデバイス (例えば、表示デバイスやプリンタなど) を含んでもよく、あるいは含まなくてもよい。また、その出力部は、ハードウェアによって実現されてもよく、あるいは、それらのデバイスを駆動するドライバ等のソフトウェアによって実現されてもよい。

【 0 2 2 6 】

なお、ここでは、コレスキー分解部 3 1 が  $L V$  型ツイスト分解法によってコレスキー分解を行う場合について説明したが、コレスキー分解部 3 1 は、 $q d$  型ツイスト分解法によってコレスキー分解を行ってもよい。例えば、コレスキー分解部 3 1 は、算出された固有値の分布を見て、分布が密でない場合には、 $q d$  型ツイスト分解法を用いるようにしてもよい。

【 0 2 2 7 】

また、上記説明では、行列  $B^{(0)}$  が上 2 重対角行列である場合について説明したが、行列  $B^{(0)}$  は下 2 重対角行列であっても固有値分解を同様に実行することができる。ただし、行列  $T$  の各要素と、行列  $B^{(0)}$  の各要素との対応を示す式が上述のものとなることになる。

【 0 2 2 8 】

また、上記説明では、固有値算出部 1 7 が全ての固有値を算出する場合について説明したが、一部の固有値のみを算出するようにしてもよい。例えば、図 9 において、固有値算出部 1 7 は、行列  $T_1, T_2$  までは、全ての固有値を算出する必要があるが、行列  $T_1, T_2$  から対称 3 重対角行列  $T$  の固有値を算出する際に、必要な範囲で固有値を算出してもよい。したがって、固有値算出部 1 7 は、対称 3 重対角行列  $T$  の少なくとも 1 個の固有値を算出するものであってもよい。その場合には、不必要な固有値を算出することに伴う余分な処理を実行しなくてよいため、処理負荷を軽減することができる。このように、一部の固有値のみを算出する場合には、例えば、対称 3 重対角行列  $T$  の固有値を算出する処理に費やす計算時間を約  $(1 + k/m)/2$  倍にすることができる。ここで、 $m$  は行列サイズであり、 $k$  は求める固有値の個数である。

【 0 2 2 9 】

また、上記説明では、固有ベクトル算出部 1 9 が算出された固有値に対応する全ての固

10

20

30

40

50

有ベクトルを算出する場合について説明したが、上記説明から明らかなように、固有ベクトルを算出する処理は、固有値ごとに行うことができる。したがって、固有ベクトル算出部 19 は、対称 3 重対角行列  $T$  の少なくとも 1 個の固有ベクトルを算出するものであってもよい。このように、固有ベクトル算出部 19 は、必要な範囲で固有ベクトルを算出することができ、不必要な固有ベクトルを算出することに伴う余分な処理を実行しなくともよい。そのため、処理負荷を軽減することができる。

#### 【0230】

次に、本実施の形態による固有値分解装置 1 における並列処理について説明する。

本実施の形態による固有値分解装置 1 では、固有値の計算及び固有ベクトルの計算において、並列的に処理を実行することができる。例えば、図 24 ~ 図 26 で示されるように、固有値分解装置 1 において、固有値分解部 15、固有値算出部 17、コレスキー分解部 22、ベクトル算出部 23、コレスキー分解部 31、ベクトル算出部 32 において、それぞれ並列処理を行ってもよい。

10

#### 【0231】

固有値分解部 15 は、複数の固有値分解手段 15 a、15 b を備え、その複数の固有値分解手段 15 a、15 b が、分割後の対称 3 重対角行列の固有値と、行列要素とを算出する処理を並列実行してもよい。例えば、図 9 において、固有値分解手段 15 a が行列  $T_{11}$ 、 $T_{112}$ 、 $T_{121}$ 、 $T_{121}$  の固有値分解を実行し、固有値分解手段 15 b が行列  $T_{211}$ 、 $T_{212}$ 、 $T_{221}$ 、 $T_{222}$  の固有値分解を実行してもよい。

20

#### 【0232】

固有値算出部 17 は、複数の固有値算出手段 17 a、17 b を備え、その複数の固有値算出手段 17 a、17 b が、分割元の対称 3 重対角行列の固有値と、行列要素とを算出する処理を並列実行してもよい。例えば、図 9 において、固有値算出手段 17 a が行列  $T_1$ 、 $T_{12}$ 、 $T_1$ 、 $T$  の固有値等を算出する処理を実行し、固有値算出手段 17 b が行列  $T_{21}$ 、 $T_{22}$ 、 $T_2$  の固有値等を算出する処理を実行してもよい。

#### 【0233】

コレスキー分解部 22 は、複数のコレスキー分解手段 22 a、22 b を備え、その複数のコレスキー分解手段 22 a、22 b が、対称 3 重対角行列  $T$  に関してコレスキー分解する処理を並列実行してもよい。例えば、コレスキー分解手段 22 a が半分の固有値についてコレスキー分解する処理を実行し、コレスキー分解手段 22 b が残りの半分の固有値についてコレスキー分解する処理を実行してもよい。

30

#### 【0234】

ベクトル算出部 23 は、複数のベクトル算出手段 23 a、23 b を備え、その複数のベクトル算出手段 23 a、23 b が、固有ベクトルを算出する処理を並列実行してもよい。例えば、ベクトル算出手段 23 a がコレスキー分解手段 22 a によってコレスキー分解された固有値に関して固有ベクトルを算出する処理を実行し、ベクトル算出手段 23 b がコレスキー分解手段 22 b によってコレスキー分解された固有値に関して固有ベクトルを算出する処理を実行してもよい。

#### 【0235】

コレスキー分解部 31 が複数のコレスキー分解手段 31 a、31 b を備えて、対称 3 重対角行列  $T$  に関してコレスキー分解する処理を並列実行してもよいことは、コレスキー分解部 22 と同様である。

40

#### 【0236】

ベクトル算出部 32 が複数のベクトル算出手段 32 a、32 b を備えて、固有ベクトルを算出する処理を並列実行してもよいことは、ベクトル算出部 23 と同様である。

#### 【0237】

なお、固有値算出部 17 の複数の固有値算出手段 17 a、17 b は、図 6 で示される分割された 2 個の行列  $T_1$ 、 $T_2$  の固有値と行列要素とから、分割元の行列  $T_0$  の固有値と行列要素とを算出する処理を、並列実行してもよい。以下、その処理について説明する。

#### 【0238】

50

まず、式 1 から、各固有値を求める処理については、並列実行可能なことがわかる。例えば、固有値算出手段 17 a が半分の固有値を算出し、固有値算出手段 17 b が残りの半分の固有値を算出してもよい。なお、固有値算出手段 17 a、17 b のそれぞれは、行列  $D_{12}$  と、ベクトル  $z$  との各成分の値を有するものとする。

【0239】

また、式 2 から、各固有ベクトル  $q_i$  を求める処理についても、並列実行可能なことがわかる。例えば、固有値算出手段 17 a は、算出した固有値に対応する半分の固有ベクトル  $q_i$  を算出し、固有値算出手段 17 b は、算出した固有値に対応する残りの半分の固有ベクトル  $q_i$  を算出してもよい。

【0240】

また、式 3、式 4 から、 $f$ 、 $l$  を求める処理についても、並列実行可能なことがわかる。例えば、固有値算出手段 17 a は、算出した固有ベクトル  $q_i$  を用いて、 $f$ 、 $l$  の一部の要素を算出し、固有値算出手段 17 b は、算出した固有ベクトル  $q_i$  を用いて、 $f$ 、 $l$  の残りの要素を算出してもよい。その後、固有値算出手段 17 a、あるいは固有値算出手段 17 b が、固有値算出手段 17 a、17 b の算出した  $f$ 、 $l$  の各要素を統合することによって、式 3、式 4 で示される  $f$ 、 $l$  を算出することができる。

【0241】

ここでは、 $z$  の成分に 0 が含まれず、 $D_{12}$  の対角成分に同じ値が含まれない場合における固有値算出手段 17 a、17 b の処理について説明したが、 $z$  の成分に 0 が含まれる場合、 $D_{12}$  の対角成分に同じ値が含まれる場合であっても、固有方程式を解く処理、固有ベクトル  $q_i$  を算出する処理、固有ベクトルから分割元の行列の行列要素を算出する処理を同様に並列実行することができる。

【0242】

上記のような各構成要素における並列処理において、各手段が固有値等の情報を格納するメモリを用いる場合に、複数の手段が同一のメモリ、すなわち共有メモリを使用してもよく、あるいは、各手段がそれぞれ別のメモリを使用してもよい。

【0243】

なお、図 24 ~ 図 26 を用いて、固有値分解部 15 や固有値算出部 17 等が 2 個の手段によって並列処理を実行する場合について説明したが、固有値分解部 15 や固有値算出部 17 等が 3 以上の手段を備え、並列処理を実行してもよい。また、固有値分解部 15、固有値算出部 17、コレスキー分解部 22、ベクトル算出部 23、コレスキー分解部 31、ベクトル算出部 32 のそれぞれにおいて並列処理が実行される場合について説明したが、いずれかの任意の 1 以上の部において、並列処理が実行されなくてもよい。例えば、固有値分解部 15 において並列処理が行われなくてもよい。

【0244】

また、その並列処理は、1 の装置において、2 以上の CPU 等を用いて実行されてもよく、2 以上の装置において実行されてもよい。例えば、図 27 で示されるように、装置 A と、装置 B とがそれぞれ固有値分解手段 15 a、15 b を備え、各装置において、固有値分解の処理が並列実行されてもよい。この場合には、装置 A の固有値分解手段 15 a を備える固有値分解部 15 - 1 と、装置 B の固有値分解手段 15 b を備える固有値分解部 15 - 2 とによって固有値分解部 15 が構成されることになる。したがって、固有値分解装置 1 は、装置 A と、装置 B とからなるシステムを構成することになる。ここでは、固有値分解部 15 の並列処理について説明したが、その他の固有値算出部 17 やコレスキー分解部 22 等についても、2 以上の装置による並列処理を行ってもよい。

【0245】

次に、数値実験による性能の評価について説明する。ここでは、本実施の形態による固有値分解装置 1 の方法 (DDC) と、標準的な分割統治法 (DC) と、QR 法 (QR) と、二分法と逆反復法とを組み合わせた方法 (B&I) とを比較した。なお、本実施の形態による固有値分解装置 1 の方法では、コレスキー分解を  $qd$  型ツイスト分解法で行い、逆反復 (図 12 のステップ S701 参照) を 1 回用いた。分割統治法としては、LAPAC

10

20

30

40

50

Kで提供されているDSTEDCを用いた。QR法としては、LAPACKで提供されているDSTEQRを用いた。二分法と逆反復法とを組み合わせた方法としては、LAPACKで提供されているDSTEVRのソースコードを一部変更したものを用いた。実験で用いるLAPACKはBLAS(Basic Linear Algebra Subprograms)を呼び出す。また、本実施の形態による固有値分解装置1の固有値分解部15では、QR法を用いて固有値分解を行った。また、この数値実験では、ランダムに与えた対角行列を、ランチョス法を用いて対称3重対角化した行列を用いた。数値実験結果のグラフにおいて、横軸は、数値実験で用いた行列の大きさである。

#### 【0246】

図28は、計算の実行時間を示す図である。実行時間が短いほど、高速に計算できることになる。図28の数値実験結果のグラフからわかるように、本実施の形態による固有値分解装置1の方法では、他の方法に比べて高速に固有値分解を行うことができることがわかる。

10

#### 【0247】

図29は、算出された固有ベクトルの直交性を示す図である。値の小さい方が、直交性が高いことになる。図29の数値実験結果のグラフからわかるように、本実施の形態による固有値分解装置1の方法では、他の方法に比べて、固有ベクトルの直交性は高くない。本実施の形態による固有値分解装置1の方法では、二分法と逆反復法とを組み合わせた方法と同程度の直交性を有することがわかる。これは、固有値をまず算出し、その後、固有ベクトルを算出していることに起因していると考えられる。

20

#### 【0248】

図30は、全固有値の相対誤差の総和を示す図である。値の小さい方が、誤差が小さく、精度が高いことになる。図30の数値実験結果のグラフからわかるように、本実施の形態による固有値分解装置1の方法では、他の方法に比べて固有値の精度が高いことがわかる。分割統治法により算出した固有値と同程度の固有値の精度を有する。本実施の形態による固有値分解装置1の方法では、分割統治法によって固有値を算出しているため、当然の結果であると考えられる。

#### 【0249】

図31は、前固有ベクトルの誤差の総和を示す図である。値の小さい方が、誤差が小さく、精度が高いことになる。図31の数値実験結果のグラフからわかるように、本実施の形態による固有値分解装置1の方法では、他の方法に比べて固有ベクトルの精度が高いことがわかる。

30

#### 【0250】

このように、本実施の形態による固有値分解装置1の方法では、高速に固有値分解をすることができ、さらに、固有値、固有ベクトルの精度の高いことがわかる。また、前述のように、並列性にも優れているため、並列処理を行うことによってさらに高速化を図ることも可能である。

#### 【0251】

[主成分分析による顔画像認識]

次に、主成分分析による顔画像認識に固有値分解を応用する場合について説明する。

40

#### 【0252】

主成分分析による顔画像認識の処理は、以下の各ステップによって行われる。

- (1) 射影行列の作成
- (2) 個人の顔画像に関する情報の登録
- (3) 入力された顔画像と、登録されている個人の顔画像に関する情報とを比較することによる認識

#### 【0253】

まず、上記(1)の射影行列の作成について説明する。

種々の人物の顔画像データを用意する。その顔画像データは、例えば、スキャナやデジタルスチルカメラ、デジタルビデオカメラ等の光学的読み取り機器によって読み取られた

50

2次元画像データであってもよい。各顔画像データは、各画素値を成分とするベクトルで表示することができる。そのベクトルを、次のようにする。ここでは、M個の顔画像データを扱うものとする。

【数72】

$$\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_M$$

【0254】

次に、その顔画像データのベクトルの平均値  $\mu$  を算出する。

【数73】

$$\mu = \frac{1}{M} \sum_{i=1}^M \mathbf{x}_i$$

10

【0255】

その算出した  $\mu$  を用いて、共分散行列  $C$  を次のように算出する。

【数74】

$$C = \frac{1}{M} \sum_{i=1}^M \sum_{j=1}^M (\mu - \mathbf{x}_i)(\mu - \mathbf{x}_j)^T$$

【0256】

その後、上述の固有値分解装置1によって、その行列  $C$  に対して、固有値分解を行う。すなわち、行列記憶部11に行列  $C$  を蓄積し、その行列  $C$  に対して対角化や、行列の分割等の処理を行うことによって、固有値分解を行う。その固有値分解された結果である固有値  $\lambda_j$  は、固有値記憶部18に蓄積されることになる。また、固有値分解された結果である固有ベクトル  $\mathbf{v}_j$  は、固有ベクトル記憶部20で記憶されている固有ベクトルに対して、対角化で用いた直交行列を作用させることによって、前述のようにして求めることができる。なお、添字  $j$  は、固有値の降順になるように設定されているものとする。すなわち、固有値  $\lambda_1$  が最大の固有値となる。

20

【0257】

その求めた固有ベクトルのうち、対応する固有値が上位から  $d$  個の固有ベクトルを用いて、射影行列  $\eta$  を次のように算出する。

30

【数75】

$$\eta = (\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_d)$$

【0258】

次に、上記(2)の個人の顔画像に関する情報の登録について説明する。

各個人について、 $m$ 個の顔画像データ

【数76】

$$\mathbf{x}_1^k, \mathbf{x}_2^k, \dots, \mathbf{x}_m^k$$

を用意する。ここで、添字  $k$  は、個人を識別するための添字である。そのベクトルに射影行列  $\eta$  をそれぞれ掛けることにより、次式のようにして  $m$ 個の  $d$ 次元ベクトル  $\xi_i^k$  を算出し、その平均値  $\mu^k$  を算出する。そして、その平均値  $\mu^k$  を登録しておく。

40

【数77】

$$\xi_i^k = \eta^T \mathbf{x}_i^k \quad i=1,2,\dots,m$$

$$\mu^k = \frac{1}{m} \sum_{i=1}^m \xi_i^k$$

【0259】

次に、上記(3)の入力された顔画像と、登録されている個人の顔画像に関する情報とを比較することによる認識について説明する。

50

ある人物の顔画像データのベクトル  $x$  が入力されたとする。すると、その顔画像データのベクトル  $x$  に射影行列 を掛けることにより、 $d$ 次元ベクトル を算出する。

【数 78】

$$\xi = \eta^T x$$

【0260】

次に、その  $d$ 次元ベクトル と、各個人の  $d$ 次元ベクトル  $\mu^k$  の平均値  $\mu^k$  との差のノルムを計算する。

【数 79】

$$\|\mu^k - \xi\|$$

10

【0261】

そのノルムがあらかじめ設定されているしきい値より小さい場合に、入力された顔画像が、添字  $k$  で識別される個人の顔画像であると判断してもよい。あるいは、すべての添字  $k$  について上記のノルムを計算し、入力された顔画像が、最小のノルムに対応する添字  $k$  で識別される個人の顔画像であると判断してもよい。このようにして、固有値分解装置 1 を用いて、主成分分析による顔画像認識を行うことができる。

【0262】

[ 2次元画像から3次元へ復元する画像処理への応用 ]

次に、物体の2次元画像から3次元へ復元する画像処理に固有値分解を応用する場合について説明する。

20

【0263】

複数の2次元画像から3次元復元を行う処理は、以下の各ステップによって行われる。

(1) 2次元画像から特徴点を抽出するステップ。

(2) 特徴点データより形状(元の物体の特徴点の3次元座標データ)及び回転(3次元データから特徴点データへの変換)に関するデータを計算するステップ。

(3) 形状及び回転のデータより可視化を行うステップ。

【0264】

以下、上記(1)、(2)の各ステップについて、図32のフローチャートを用いて説明する。ここで、2次元画像は、例えば、スキャナやデジタルスチルカメラ、デジタルビデオカメラ等の光学的読み取り機器によって読み取られた2次元画像であってもよい。

30

【0265】

ステップ S5001において、2次元画像  $j$  ( $j = 1, \dots, m$ ,  $m$  は3以上の整数) から特徴点  $i$  ( $i = 1, \dots, n$ ,  $n$  は2以上の整数) の座標  $(x_i^j, y_i^j)$  を抽出する。取り扱う2次元画像は、弱中心射影画像であることが好ましい。このとき、次の式が成り立つ。

【数 80】

$$\begin{pmatrix} x_i^j \\ y_i^j \end{pmatrix} = s_j \begin{pmatrix} \mathbf{r}_{1,j}^T \\ \mathbf{r}_{2,j}^T \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix}$$

40

【0266】

ここで、 $s_j$  は物体のスケールに相対する  $j$  番目の画像のスケール、 $\mathbf{r}_{1,j}$ ,  $\mathbf{r}_{2,j}$  はそれぞれ物体座標系に相対する  $j$  番目のカメラ座標系の回転行列の1番目と2番目の行ベクトル、 $(X_i, Y_i, Z_i)^T$  は  $i$  番目の点の3次元座標である。物体のスケールは1番目の画像のスケールと同じにし ( $s_1 = 1$ )、物体の座標系の姿勢は1番目の画像のカメラ座標系と同じにする ( $\mathbf{r}_{1,1} = (1, 0, 0)^T$ ,  $\mathbf{r}_{2,1} = (0, 1, 0)^T$ )。  $m$  枚全ての画像における  $n$  個全ての点の座標を行列  $D$  の要素として並べると、次式のようなになる。

【0267】

50

【数 8 1】

$$D = MS$$

$$\text{ただし、} D = \begin{pmatrix} x_1^1 & \cdots & x_i^1 & \cdots & x_n^1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_1^j & \cdots & x_i^j & \cdots & x_n^j \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_1^m & \cdots & x_i^m & \cdots & x_n^m \\ y_1^1 & \cdots & y_i^1 & \cdots & y_n^1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ y_1^j & \cdots & y_i^j & \cdots & y_n^j \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ y_1^m & \cdots & y_i^m & \cdots & y_n^m \end{pmatrix}, \quad M = \begin{pmatrix} s_1 \mathbf{r}_{1,1}^T \\ \cdots \\ s_j \mathbf{r}_{1,j}^T \\ \cdots \\ s_m \mathbf{r}_{1,m}^T \\ s_1 \mathbf{r}_{2,1}^T \\ \cdots \\ s_j \mathbf{r}_{2,j}^T \\ \cdots \\ s_m \mathbf{r}_{2,m}^T \end{pmatrix},$$

10

$$S = \begin{pmatrix} X_1 & \cdots & X_i & \cdots & X_n \\ Y_1 & \cdots & Y_i & \cdots & Y_n \\ Z_1 & \cdots & Z_i & \cdots & Z_n \end{pmatrix}$$

【0 2 6 8】

20

MとSの形から分かるように、Dのランクは3である。ここで、ステップS 5 0 0 1において、行列Dが与えられている。以下、回転に関するデータM及び形状Sを求める。

【0 2 6 9】

そこで、行列Dの特異値分解

$$D = U \quad V^T$$

を考える。ここで、 $\sigma_j$  は特異値を大小順に対角線上に並べたもので、U及びVはそれぞれ左直交行列、及び右直交行列である。この特異値分解において、前述の固有値分解を用いることができる。すなわち、固有値分解装置1において、行列記憶部11が記憶している対称行列Aを、 $D^T D$ とすることにより、前述の説明のようにして、対称行列Aの固有値分解がなされることになる。なお、固有値分解装置1において、固有ベクトル記憶部20に蓄積されるのは、対称行列 $D^T D$ を変換した対称3重対角行列Tの固有ベクトルであるので、その固有ベクトルを、前述の説明のようにして、行列 $D^T D$ の固有ベクトルに変換する必要がある。また、その行列 $D^T D$ の各固有値の $1/2$ 乗(平方根)が各特異値となる。すなわち、特異値 $\sigma_j = (\lambda_j)^{1/2}$ となる。また、固有ベクトルと、右直交行列Vの各列を構成する右特異ベクトル $x_j$ とは等しいため、固有ベクトルを各列に有する行列が、右直交行列Vとなる。なお、特異値 $\sigma_j$ が0でない場合には、左直交行列Uの各列を構成する左特異ベクトル $y_j$ を次のようにして求めることができる。

30

【0 2 7 0】

【数 8 2】

$$\mathbf{y}_j = \frac{D \mathbf{x}_j}{\sigma_j}$$

40

となる。一方、特異値 $\sigma_j$ が0である場合には、

【数 8 3】

$$D^T \mathbf{y}_j = 0$$

を解くことにより、 $\mathbf{y}_j$ を算出することができる。ここで、次の置き換えを行うことによって、左特異ベクトル $\mathbf{y}_j$ を正規化する。

【数 8 4】

$$y_j \leftarrow \frac{y_j}{\|y_j\|}$$

【0 2 7 1】

このようにして、左特異ベクトル  $y_j$  を求めることができ、この左特異ベクトル  $y_j$  を用いて、 $U = (y_1, y_2, \dots, y_m)$  とすることにより、左直交行列  $U$  を算出することができる。

【0 2 7 2】

ここで、画像のデジタル誤差のため、ゼロでない特異値は3つ以上出てくる。しかし、4番目以降の特異値はノイズによるもので、最初の3つの特異値と比べて格段に小さい。そこで、ステップ S 5 0 0 2 では、最初の3つの特異値に対して特異ベクトルを計算する。すなわち、本実施の形態による固有値分解装置 1 では、最初の3個の固有値に対して固有ベクトルを計算する。採用する3個のベクトルをまとめると、次式となる。

10

【0 2 7 3】

$$D' = L' \quad R'^T = M' S'$$

ただし、 $M' = L' (\quad)^{1/2}$ 、 $S' = (\quad)^{1/2} R'^T$ 、 $D'$  は  $D - D'$  を最小にするランク 3 の行列である。

【0 2 7 4】

次に、 $D'$  から  $M$  及び  $S$  を求めたいが、その組合せは唯一ではない。なぜなら、任意の正則行列  $C$  が

20

$$D' = (M' C) (C^{-1} S')$$

を満たすからである。そこで、上式における  $C$  を  $M = M' C$  を満たすように決める。 $C$  は下記の式を満たす。

【数 8 5】

$$MM^T = M'CC^TM'^T = \begin{pmatrix} 1 & \dots & \dots & 0 & \dots & \dots \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \dots & \dots & s_m^2 & \dots & \dots & 0 \\ 0 & \dots & \dots & 1 & \dots & \dots \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \dots & \dots & 0 & \dots & \dots & s_m^2 \end{pmatrix}$$

30

【0 2 7 5】

$E = C C^T$  とすると、上式から  $E$  の6つの要素に関する  $2m + 1$  個の線形方程式が得られる。 $m = 3$  であるので、 $E$  の要素を一意に決めることができる。ステップ S 5 0 0 3 において、行列  $E$  を求める。

【0 2 7 6】

次に、ステップ S 5 0 0 4 において、 $E$  から  $C$  を求める。 $C$  の自由度 (9) は  $E$  の自由度 (6) より多い。そこで、条件  $r_{1j} = (1, 0, 0)^T$ 、 $r_{2j} = (0, 1, 0)^T$  を加えれば、 $C$  を決めることができる。このとき2つの解 (Necker Reversal) が出る。

40

次に、ステップ S 5 0 0 5 において、 $M = M' C$  及び  $S = C^{-1} S'$  より、回転に関するデータ  $M$  及び形状  $S$  が決まる。

【0 2 7 7】

[ 文書検索への応用 ]

次に、文書検索処理に固有値分解を応用する場合について説明する。文書中からその文書の内容に関連する索引語を抽出し、索引語の重みを計算する処理の後、ベクトル空間モデルでは、この索引語の重みを要素とするベクトルで文書を表現する。ここで、検索対象となる文書を  $d_1, d_2, \dots, d_n$  とし、これら文書集合全体を通して全部で  $m$  個の

50



索引語  $w_1, w_2, \dots, w_m$  があるとする。このとき、文書  $d_j$  は、次のようなベクトルで表現されることになる。これを文書ベクトルと呼ぶ。

【数 8 6】

$$\mathbf{d}_j = \begin{pmatrix} d_{1j} \\ d_{2j} \\ \vdots \\ d_{mj} \end{pmatrix}$$

【0 2 7 8】

ここで、 $d_{ij}$  は索引語  $w_i$  の文書  $d_j$  における重みである。また、文書集合全体は、

10

次のような  $m \times n$  行列  $D$  によって表現することができる。

【数 8 7】

$$D = (\mathbf{d}_1 \quad \mathbf{d}_2 \quad \dots \quad \mathbf{d}_n) = \begin{pmatrix} d_{11} & d_{12} & \dots & d_{1n} \\ d_{21} & d_{22} & \dots & d_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{m1} & d_{m2} & \dots & d_{mn} \end{pmatrix}$$

【0 2 7 9】

行列  $D$  を索引語文書行列と呼ぶ。索引語文書行列の各列は文書に関する情報を表している文書ベクトルであるが、同様に、索引語文書行列の各行は索引語に関する情報を表しているベクトルであり、これを索引語ベクトルと呼ぶ。検索質問も、文書と同様に、索引語の重みを要素とするベクトルで表現することができる。検索質問文に含まれる索引語  $w_i$  の重みを  $q_i$  とすると、検索質問ベクトル  $\mathbf{q}$  は次のように表されることになる。

20

【数 8 8】

$$\mathbf{q} = \begin{pmatrix} q_1 \\ q_2 \\ \vdots \\ q_m \end{pmatrix}$$

【0 2 8 0】

実際の文書検索においては、与えられた検索質問文と類似した文書を見つけ出す必要があるが、検索質問ベクトル  $\mathbf{q}$  と各文書ベクトル  $\mathbf{d}_j$  の間の類似度を計算することにより行う。ベクトル間の類似度の定義としてはさまざまなものが考えられるが、文書検索においてよく用いられているものはコサイン尺度（2つのベクトルのなす角度）または内積である。

30

【0 2 8 1】

【数 8 9】

$$\cos(\mathbf{d}_j, \mathbf{q}) = \frac{\mathbf{d}_j \cdot \mathbf{q}}{\|\mathbf{d}_j\| \|\mathbf{q}\|} = \frac{\sum_{i=1}^m d_{ij} q_i}{\sqrt{\sum_{i=1}^m d_{ij}^2} \sqrt{\sum_{i=1}^m q_i^2}} \quad (\text{コサイン尺度})$$

40

$$\mathbf{d}_j \cdot \mathbf{q} = \sum_{i=1}^m d_{ij} q_i \quad (\text{内積})$$

なお、ベクトルの長さが 1 に正規化（コサイン正規化）されている場合には、コサイン尺度と内積とは一致する。

【0 2 8 2】

図 3 3 は、本実施の形態による固有値分解装置 1 を利用した文書検索方法の一例を示すフローチャートである。

50

ステップ S 6 0 0 1 において、質問ベクトル  $q$  を受け取る。

【 0 2 8 3 】

ここでは、 $D$  の近似行列  $D_k$  を使った検索を考える。ベクトル空間モデルでは、検索質問ベクトル  $q$  と索引語文書行列  $D$  中の各文書ベクトル  $d_j$  の間の類似度を計算することにより検索を行うが、ここでは  $D$  の代わりに  $D_k$  を使う。ベクトル空間モデルでは、文書ベクトルの次元数は索引語の総数と等しい。したがって、検索対象となる文書の数が増えるに従い、文書ベクトルの次元数も増加する傾向にある。しかし、次元数が増加してくると、コンピュータのメモリによる制限や検索時間の増大などの問題が生じてくるばかりでなく、文書中に含まれる不必要な索引語がノイズ的な影響を及ぼし、検索精度を低下させてしまうという現象も起こってくる。潜在的意味インデキシング (latent semantic indexing; LSI) は、高次元の空間にある文書ベクトルを低次元の空間へと射影することにより、検索精度の改善を図る技術である。高次元の空間では別々に扱われていた索引語が、低次元の空間では相互に関連を持ったものとして扱われる可能性もあるため、索引語の持つ意味や概念に基づく検索を行うことができる。たとえば、通常のベクトル空間モデルでは "car" という索引語と "automobile" という索引語はまったく別物であり、一方の索引語による質問ではもう片方の索引語を含んだ文書を検索することができない。しかし、低次元の空間ではこれらの意味的に関連した索引語は 1 つの次元に縮退することが期待できるため、"car" という検索質問によって "car" を含む文書ばかりでなく "automobile" を含む文書をも検索することが可能となる。潜在的意味インデキシングでは、特異値分解により高次元ベクトルの次元削減を行うが、これは基本的に多変量解析における主成分分析と等価である。

10

20

【 0 2 8 4 】

ステップ S 6 0 0 2 において、 $k$  を選択する。ここでは、 $k < r$  なる  $k$  の値を選択する。 $r = \min(n, m)$  である。 $k$  の値は、予め与えられていてもよいし、計算ごとに選択可能であってもよい。

【 0 2 8 5 】

次に、ステップ S 6 0 0 3 では、行列  $D$  の特異値分解を行う。この特異値分解として、前述の 2 次元画像から 3 次元へ復元する画像処理への応用で説明したように、固有値分解による方法を用いることができる。すなわち、固有値分解装置 1 において、行列記憶部 11 が記憶している対称行列  $A$  を、このたびの行列  $D^T D$  とすることにより、前述の説明のようにして、対称行列  $A$  の固有値分解がなされ、その結果として、行列  $D$  の特異値分解を行うことができる。なお、この特異値分解では、計算された特異値のうち、大きい順に 1 番目から  $k$  番目までの  $k$  個の特異値に対して  $D$  の特異ベクトルを算出する。すなわち、固有値分解において、計算された固有値のうち、大きい順に 1 番目から  $k$  番目までの  $k$  個の固有値に対して固有ベクトルを算出し、この固有ベクトルを用いて特異ベクトルを算出すればよい。 $k$  は、ステップ S 6 0 0 2 で選択された値である。

30

【 0 2 8 6 】

すなわち、

$$D_k = U_k V_k^T$$

なる  $U_k$  及び  $V_k$  を計算する。ここで、 $U_k$  は、最初の  $k$  個の左特異ベクトルのみから構成される  $m \times k$  行列であり、 $V_k$  は、最初の  $k$  個の右特異ベクトルのみから構成される  $n \times k$  行列であり、 $D_k$  は、最初の  $k$  個の特異値のみから構成される  $k \times k$  対角行列である。

40

【 0 2 8 7 】

次に、ステップ S 6 0 0 4 において、行列  $D_k$  と質問ベクトル  $q$  との類似度を計算する。いま、ベクトル  $e_j$  を  $n$  次元の単位ベクトルとすると、 $D_k$  の  $j$  番目の文書ベクトルは  $D_k e_j$  で表すことができる。文書ベクトル  $D_k e_j$  と検索質問ベクトル  $q$  との間の類似度計算は、

【数 9 0】

$$\begin{aligned} \cos(D_k \mathbf{e}_j, \mathbf{q}) &= \frac{(D_k \mathbf{e}_j) \cdot \mathbf{q}}{\|D_k \mathbf{e}_j\| \|\mathbf{q}\|} = \frac{(D_k \mathbf{e}_j)^T \mathbf{q}}{\|D_k \mathbf{e}_j\| \|\mathbf{q}\|} \\ &= \frac{(U_k \Sigma_k V_k^T \mathbf{e}_j)^T \mathbf{q}}{\|U_k \Sigma_k V_k^T \mathbf{e}_j\| \|\mathbf{q}\|} = \frac{\mathbf{e}_j^T V_k \Sigma_k U_k^T \mathbf{q}}{\|\Sigma_k V_k^T \mathbf{e}_j\| \|\mathbf{q}\|} \\ &= \frac{(\Sigma_k V_k^T \mathbf{e}_j)^T (U_k^T \mathbf{q})}{\|\Sigma_k V_k^T \mathbf{e}_j\| \|\mathbf{q}\|} \end{aligned}$$

10

としてもよいが、別の定義を用いてもよい。上式では、 $D_k$  を  $U_k$  ,  $\Sigma_k$  ,  $V_k$  から再構成する必要はなく特異値分解の結果から、直接、類似度を計算できることを示している。

上式の中に現われる  $\Sigma_k V_k^T \mathbf{e}_j$  は、

$$\Sigma_k V_k^T \mathbf{e}_j = U_k^T D_k \mathbf{e}_j$$

と書き直すことができる。この式の右辺は、近似行列  $D_k$  における  $j$  番目の文書ベクトルの基底  $U_k$  のもとでの座標（文書の  $k$  次元表現）を表している。同様に、上式の中の  $U_k^T \mathbf{q}$  は、検索質問ベクトル  $\mathbf{q}$  の基底  $U_k$  のもとでの座標（検索質問の  $k$  次元表現）である。

【0288】

ステップ S 6 0 0 5 において、ステップ S 6 0 0 4 において計算された類似度を基準に、検索結果を出力する。

20

このように、本実施の形態による固有値分解装置 1 は、画像処理や検索処理等の種々の処理に対して用いることができる。なお、上記以外の処理に用いてもよいことは言うまでもない。

【0289】

以上のように、本実施の形態による固有値分解装置 1 では、分割統治法を用いて固有値のみを算出するため、標準的な分割統治法で実行されるベクトル更新が必要なくなり、標準的な分割統治法よりも非常に高速である。また、固有値から固有ベクトルを算出する処理も、高速に処理することができる。さらに、QR法では固有値を算出する段階の並列化が困難であるが、本実施の形態による固有値分解装置 1 では、固有値を算出する段階、及び固有値から固有ベクトルを算出する段階のそれぞれにおいて、本質的に高い並列性を持つ。また、本実施の形態による固有値分解装置 1 では、他の方法と比べて高い精度の得られることがわかる。

30

【0290】

なお、本実施の形態による固有値分解装置 1 は、スタンドアロンの装置であってもよく、サーバクライアントシステムにおけるサーバ装置であってもよく、前述のように、複数の装置から構成されるシステムであってもよい。固有値分解装置 1 がサーバクライアントシステムにおけるサーバ装置である場合には、行列記憶部 11 で記憶される対称行列  $A$  や、固有値記憶部 18 で記憶される固有値や固有ベクトル記憶部 20 で記憶される固有ベクトル等は、インターネットやイントラネット等の通信回線を介して送受信されてもよい。

40

【0291】

また、本実施の形態による固有値分解装置 1 における固有ベクトルの算出において、ある固有値を用いた固有ベクトルの算出では、 $q d$  型ツイスト分解法を用い、他の固有値を用いた固有ベクトルの算出では、 $L V$  型ツイスト分解法を用いるようにしてもよい。例えば、値の近接している固有値については、 $L V$  型ツイスト分解法を用いて固有ベクトルの算出を行い、値が近接していない固有値については、 $q d$  型ツイスト分解法を用いて固有ベクトルの算出を行ってもよい。固有値の値が近接しているかどうかは、所定の記録媒体等においてあらかじめ設定されているしきい値と比較することによって判断してもよい。

【0292】

また、上記実施の形態において、各処理または各機能は、単一の装置または単一のシス

50

テムによって集中処理されることによって実現されてもよく、あるいは、複数の装置または複数のシステムによって分散処理されることによって実現されてもよい。

【0293】

また、上記実施の形態において、各構成要素は専用のハードウェアにより構成されてもよく、あるいは、ソフトウェアにより実現可能な構成要素については、プログラムを実行することによって実現されてもよい。例えば、ハードディスクや半導体メモリ等の記録媒体に記録されたソフトウェア・プログラムをCPU等のプログラム実行部が読み出して実行することによって、各構成要素が実現され得る。なお、上記実施の形態における固有値分解装置を実現するソフトウェアは、以下のようなプログラムである。つまり、このプログラムは、コンピュータに、対称3重対角行列Tが記憶される対角行列記憶部から前記対称3重対角行列Tを読み出し、当該対称3重対角行列Tを2個の対称3重対角行列に分割して前記対角行列記憶部に蓄積し、その対称3重対角行列を2個の対称3重対角行列に分割して前記対角行列記憶部に蓄積する処理を、分割後の各対称3重対角行列があらかじめ決められた大きさ以下となるまで繰り返す行列分割ステップと、前記あらかじめ決められた大きさ以下の各対称3重対角行列を前記対角行列記憶部から読み出し、前記各対称3重対角行列に対して固有値分解を行い、前記各対称3重対角行列の固有値と、前記各対称3重対角行列の固有ベクトルからなる直交行列の行列要素とを少なくとも算出し、当該固有値と当該行列要素とを固有値分解記憶部に蓄積する固有値分解ステップと、各対称3重対角行列の固有値と、行列要素とを前記固有値分解記憶部から読み出し、前記固有値と、前記行列要素とから、分割元の対称3重対角行列の固有値と、分割元の対称3重対角行列の行列要素とを算出して前記固有値分解記憶部に蓄積し、その分割元の対称3重対角行列の固有値と、行列要素とを算出する処理を、対称3重対角行列Tの少なくとも1個の固有値を算出するまで繰り返し、前記対称3重対角行列Tの少なくとも1個の固有値を固有値記憶部に蓄積する固有値算出ステップと、前記対角行列記憶部から前記対称3重対角行列Tを読み出し、前記固有値記憶部から前記対称3重対角行列Tの固有値を読み出し、前記対称3重対角行列Tとその固有値とから、ツイスト分解法を用いて前記対称3重対角行列Tの少なくとも1個の固有ベクトルを算出して固有ベクトル記憶部に蓄積する固有ベクトル算出ステップと、を実行させるためのものである。

また、このプログラムでは、前記固有ベクトル算出ステップにおいて、qd型ツイスト分解法により固有ベクトルを算出してもよい。

【0294】

また、このプログラムでは、前記固有ベクトル算出ステップが、前記固有値記憶部から前記対称3重対角行列Tの固有値を読み出し、当該固有値のいずれかが負の値である場合に、前記対角行列記憶部から前記対称3重対角行列Tを読み出し、当該対称3重対角行列Tの固有ベクトルを変化させることなく、当該対称3重対角行列Tのすべての固有値が正の値となるように、当該対称3重対角行列Tを正定値化し、正定値化した後の固有値を前記固有値記憶部に蓄積し、正定値化した後の対称3重対角行列Tを前記対角行列記憶部に蓄積する正定値化ステップと、前記対角行列記憶部からすべての固有値が正の値である対称3重対角行列Tを読み出し、前記固有値記憶部から前記対称3重対角行列Tの正の値の固有値を読み出し、前記対称3重対角行列Tの各要素に関してqd型変換を行うことによって、前記対称3重対角行列Tを上2重対角行列 $B^{(+1)}$ 及び下2重対角行列 $B^{(-1)}$ にコレスキー分解するコレスキー分解ステップと、前記上2重対角行列 $B^{(+1)}$ 及び下2重対角行列 $B^{(-1)}$ の各要素と、前記対称3重対角行列Tの正の値の固有値とを用いて固有ベクトルを算出して前記固有ベクトル記憶部に蓄積するベクトル算出ステップと、を備えてもよい。

また、このプログラムでは、前記固有ベクトル算出ステップにおいて、LV型ツイスト分解法により固有ベクトルを算出してもよい。

【0295】

また、このプログラムでは、前記固有ベクトル算出ステップが、前記対角行列記憶部から前記対称3重対角行列Tを読み出し、前記固有値記憶部から前記対称3重対角行列Tの

固有値を読み出し、前記対称3重対角行列Tの各要素に関してミウラ変換、dLVv型変換、逆ミウラ変換を行うことによって、前記対称3重対角行列Tを上2重対角行列 $B^{(+)}$ 及び下2重対角行列 $B^{(-)}$ にコレスキー分解するコレスキー分解ステップと、前記上2重対角行列 $B^{(+)}$ 及び下2重対角行列 $B^{(-)}$ の各要素と、前記対称3重対角行列Tの固有値とを用いて固有ベクトルを算出して前記固有ベクトル記憶部に蓄積するベクトル算出ステップと、を備えてもよい。

【0296】

また、このプログラムでは、コンピュータに、対称行列Aが記憶される行列記憶部から前記対称行列Aを読み出し、前記対称行列Aを3重対角化した前記対称3重対角行列Tを算出して前記対角行列記憶部に蓄積する対角化ステップをさらに実行させてもよい。

10

なお、上記プログラムにおいて、情報を蓄積するステップなどでは、ハードウェアで行われたい処理は少なくとも含まれない。

【0297】

また、このプログラムは、サーバなどからダウンロードされることによって実行されてもよく、所定の記録媒体（例えば、CD-ROMなどの光ディスクや磁気ディスク、半導体メモリなど）に記録されたプログラムが読み出されることによって実行されてもよい。

【0298】

また、このプログラムを実行するコンピュータは、単数であってもよく、複数であってもよい。すなわち、集中処理を行ってもよく、あるいは分散処理を行ってもよい。

20

【0299】

図34は、上記プログラムを実行して、上記実施の形態による固有値分解装置1を実現するコンピュータの外観の一例を示す模式図である。上記実施の形態は、コンピュータハードウェア及びその上で実行されるコンピュータプログラムによって実現される。

【0300】

図34において、コンピュータシステム100は、CD-ROM (Compact Disk Read Only Memory) ドライブ105、FD (Flexible Disk) ドライブ106を含むコンピュータ101と、キーボード102と、マウス103と、モニタ104とを備える。

【0301】

図35は、コンピュータシステムを示す図である。図35において、コンピュータ101は、CD-ROMドライブ105、FDドライブ106に加えて、CPU (Central Processing Unit) 111と、ブートアッププログラム等のプログラムを記憶するためのROM (Read Only Memory) 112と、CPU 111に接続され、アプリケーションプログラムの命令を一時的に記憶すると共に、一時記憶空間を提供するRAM (Random Access Memory) 113と、アプリケーションプログラム、システムプログラム、及びデータを記憶するハードディスク114と、CPU 111、ROM 112等を相互に接続するバス115とを備える。なお、コンピュータ101は、LANへの接続を提供する図示しないネットワークカードを含んでもよい。

30

【0302】

コンピュータシステム100に、上記実施の形態による固有値分解装置1の機能を実行させるプログラムは、CD-ROM 121、またはFD 122に記憶されて、CD-ROMドライブ105、またはFDドライブ106に挿入され、ハードディスク114に転送されてもよい。これに代えて、そのプログラムは、図示しないネットワークを介してコンピュータ101に送信され、ハードディスク114に記憶されてもよい。プログラムは実行の際にRAM 113にロードされる。なお、プログラムは、CD-ROM 121やFD 122、またはネットワークから直接、ロードされてもよい。

40

【0303】

また、行列記憶部11、対角行列記憶部13、固有値分解記憶部16、固有値記憶部18、固有ベクトル記憶部20は、RAM 113やハードディスク114によって実現され

50

てもよい。

【0304】

プログラムは、コンピュータ101に、上記実施の形態による固有値分解装置1の機能を実行させるオペレーティングシステム(OS)、またはサードパーティプログラム等を必ずしも含まなくてもよい。プログラムは、制御された態様で適切な機能(モジュール)を呼び出し、所望の結果が得られるようにする命令の部分のみを含んでいてもよい。コンピュータシステム100がどのように動作するのかについては周知であり、詳細な説明を省略する。

また、本発明は、以上の実施の形態に限定されることなく、種々の変更が可能であり、それらも本発明の範囲内に包含されるものであることは言うまでもない。

10

また、本発明のほんのいくつかの典型的な実施例について上で詳細に説明したが、その典型的な実施例において、発明の利益と新規な技術から実質的にはずれることなく多くの変更が可能であることを当業者は容易に認識することができるであろう。したがって、そのようなすべての変更は、本発明の範囲に含まれるものである。

【産業上の利用可能性】

【0305】

以上のように、本発明による固有値分解装置等によれば、固有値分解を高速に処理することができ、分子軌道法による化学計算処理や統計計算処理、情報検索処理、その他の固有値分解を用いる処理を実行する装置等において有用である。

【図面の簡単な説明】

20

【0306】

【図1】本発明の実施の形態1による固有値分解装置の構成を示すブロック図

【図2】同実施の形態による固有値分解装置の動作を示すフローチャート

【図3】同実施の形態による固有値分解装置の動作を示すフローチャート

【図4】同実施の形態による固有値分解装置の動作を示すフローチャート

【図5】同実施の形態における対称3重対角行列Tの分割について説明するための図

【図6】同実施の形態における対称3重対角行列Tの固有値の算出について説明するための図

【図7】同実施の形態による固有値分解装置の動作を示すフローチャート

【図8】同実施の形態による固有値分解装置の動作を示すフローチャート

30

【図9】同実施の形態における対称3重対角行列Tの固有値の算出について説明するための図

【図10】同実施の形態における固有ベクトル算出部の構成を示すブロック図

【図11】同実施の形態による固有値分解装置の動作を示すフローチャート

【図12】同実施の形態による固有値分解装置の動作を示すフローチャート

【図13】同実施の形態における固有ベクトル算出部の構成を示すブロック図

【図14】同実施の形態による固有値分解装置の動作を示すフローチャート

【図15】同実施の形態におけるコレスキー分解について説明するための図

【図16】同実施の形態による固有値分解装置の動作を示すフローチャート

【図17】同実施の形態による固有値分解装置の動作を示すフローチャート

40

【図18】同実施の形態による固有値分解装置の動作を示すフローチャート

【図19】同実施の形態による固有値分解装置の動作を示すフローチャート

【図20】同実施の形態による固有値分解装置の動作を示すフローチャート

【図21】同実施の形態による固有値分解装置の動作を示すフローチャート

【図22】同実施の形態におけるqd型ツイスト分解法とLV型ツイスト分解法とについて説明するための図

【図23】同実施の形態による固有値分解装置の動作を示すフローチャート

【図24】同実施の形態による固有値分解装置の構成の他の一例を示すブロック図

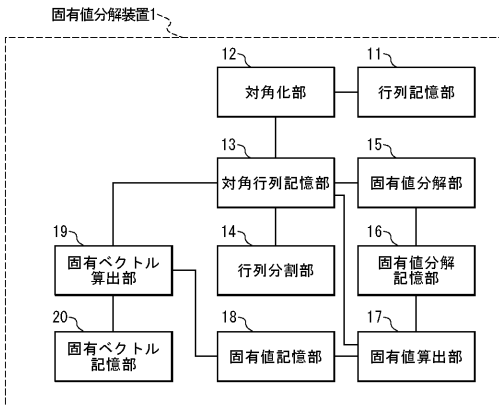
【図25】同実施の形態における固有ベクトル算出部の構成の他の一例を示すブロック図

【図26】同実施の形態における固有ベクトル算出部の構成の他の一例を示すブロック図

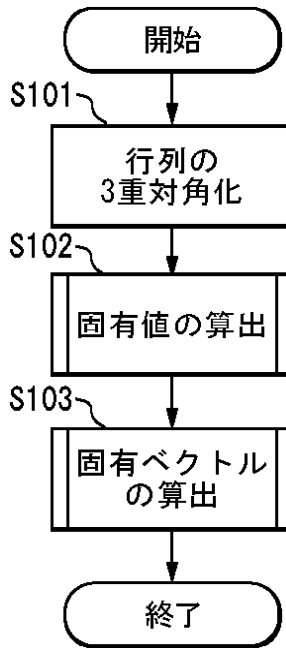
50

- 【図27】同実施の形態による固有値分解装置の構成の他の一例を示すブロック図
- 【図28】同実施の形態による固有値分解装置の数値実験の結果を示す図
- 【図29】同実施の形態による固有値分解装置の数値実験の結果を示す図
- 【図30】同実施の形態による固有値分解装置の数値実験の結果を示す図
- 【図31】同実施の形態による固有値分解装置の数値実験の結果を示す図
- 【図32】同実施の形態における画像処理の一例を示すフローチャート
- 【図33】同実施の形態における文書検索処理の一例を示すフローチャート
- 【図34】同実施の形態におけるコンピュータシステムの外觀一例を示す模式図
- 【図35】同実施の形態におけるコンピュータシステムの構成の一例を示す図

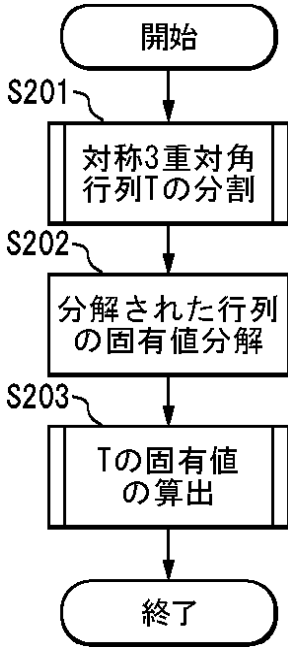
【図1】



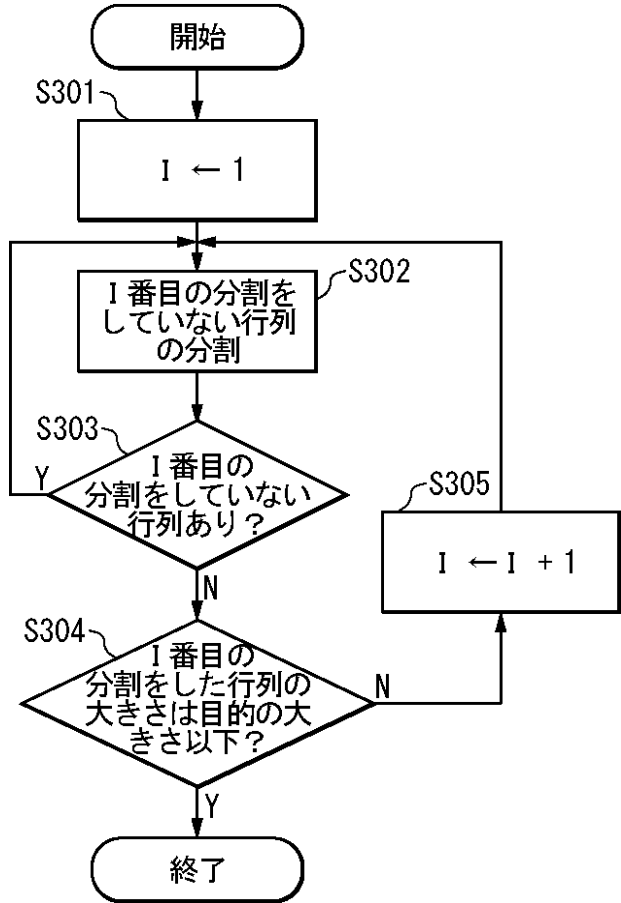
【図2】



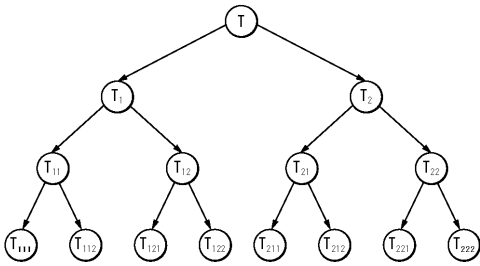
【図3】



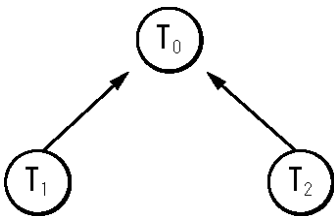
【図4】



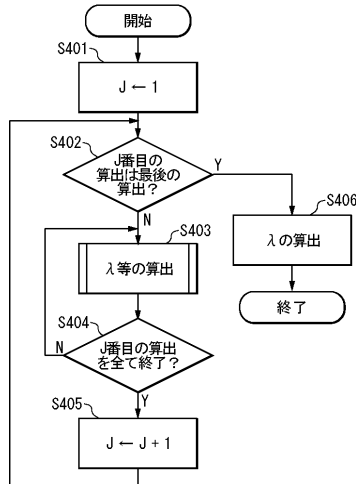
【図5】



【図6】

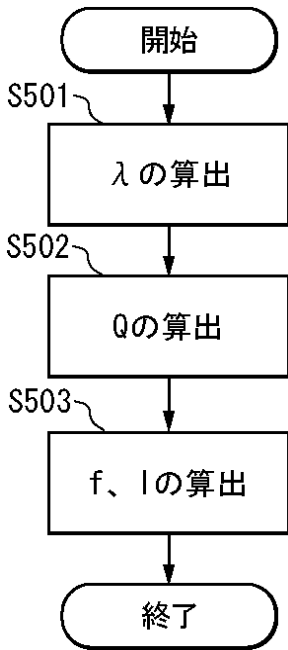


【図7】

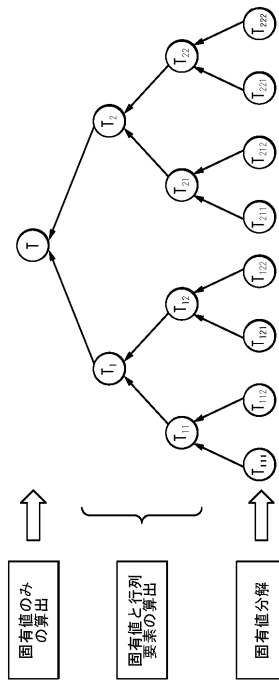




【図8】

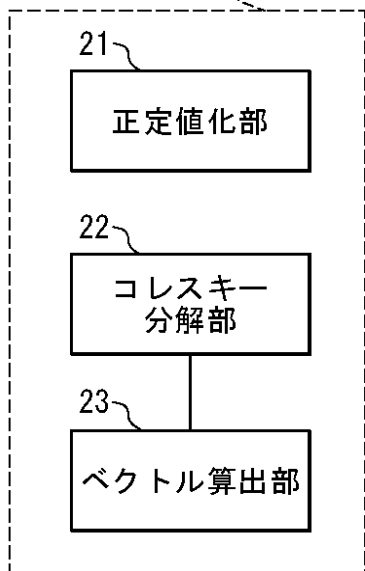


【図9】

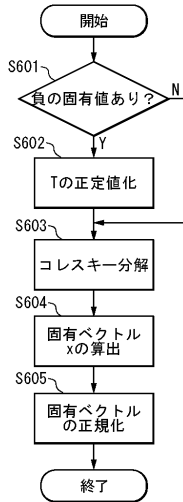


【図10】

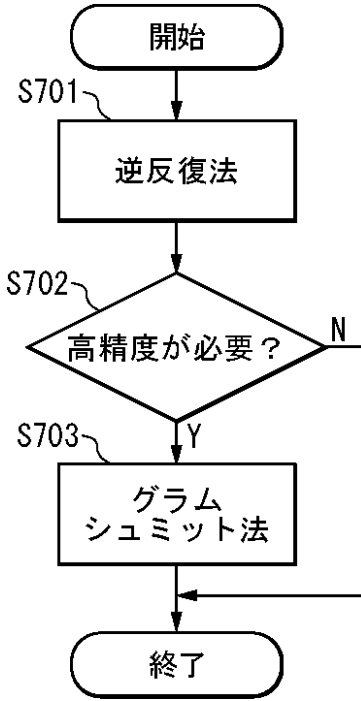
固有ベクトル算出部19



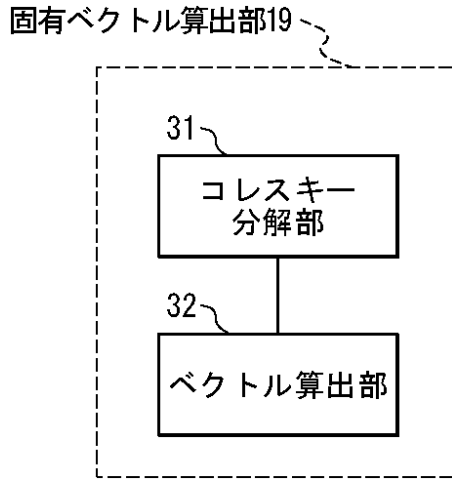
【図11】



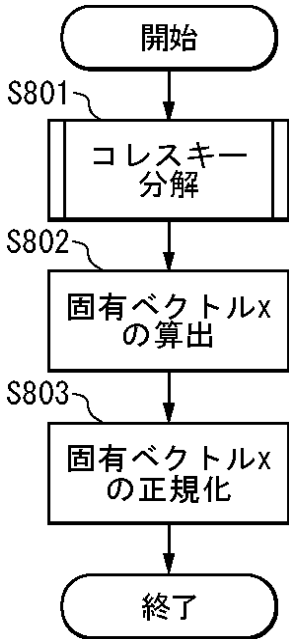
【図12】



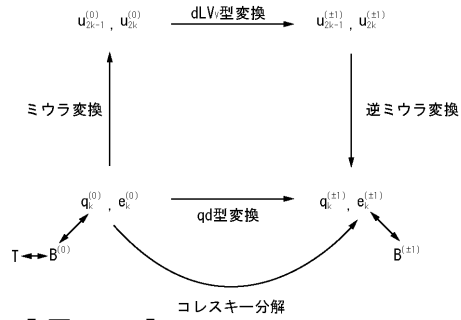
【図13】



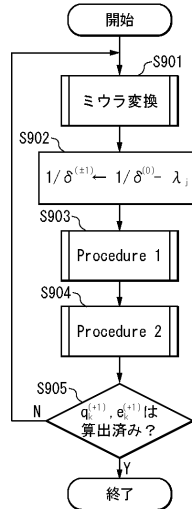
【図14】



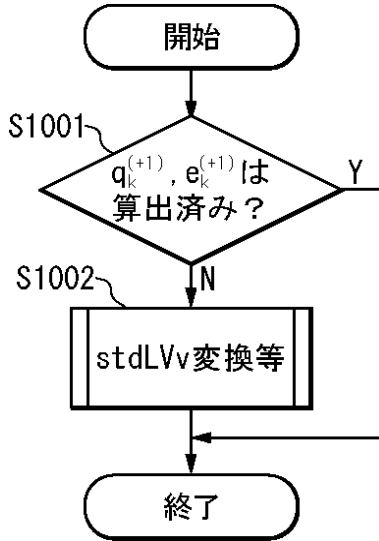
【図15】



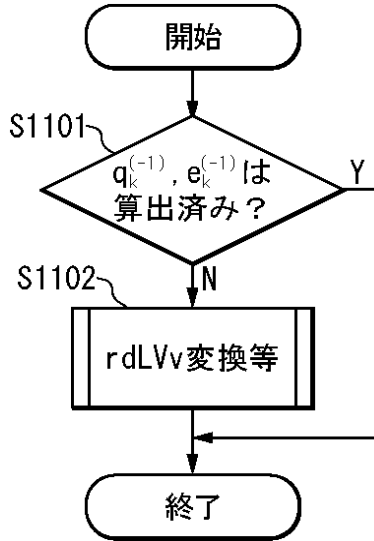
【図16】



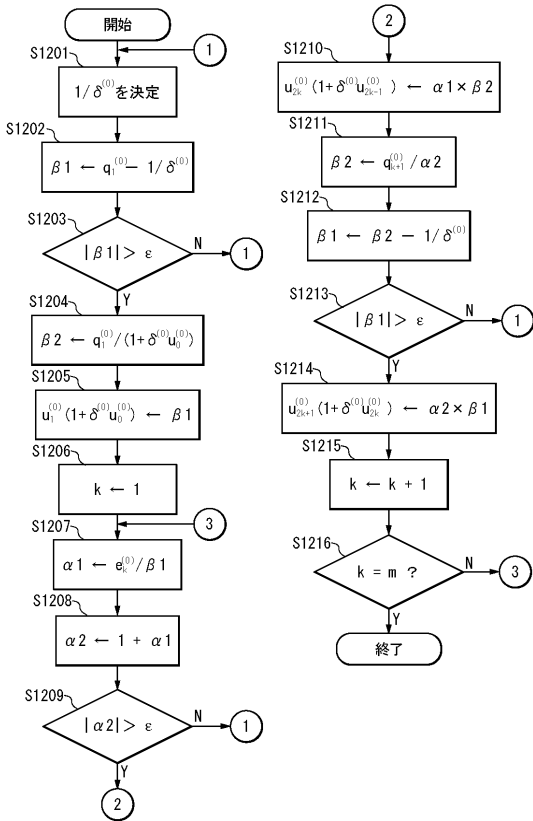
【図17】



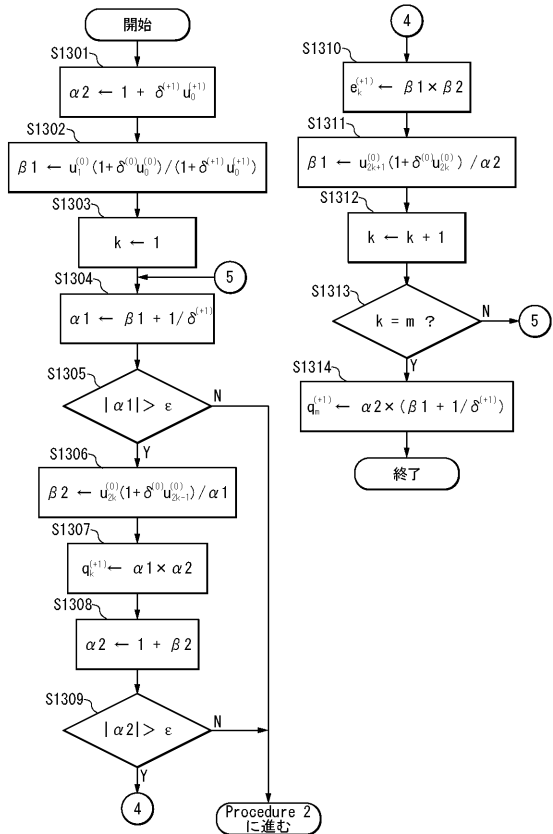
【図18】



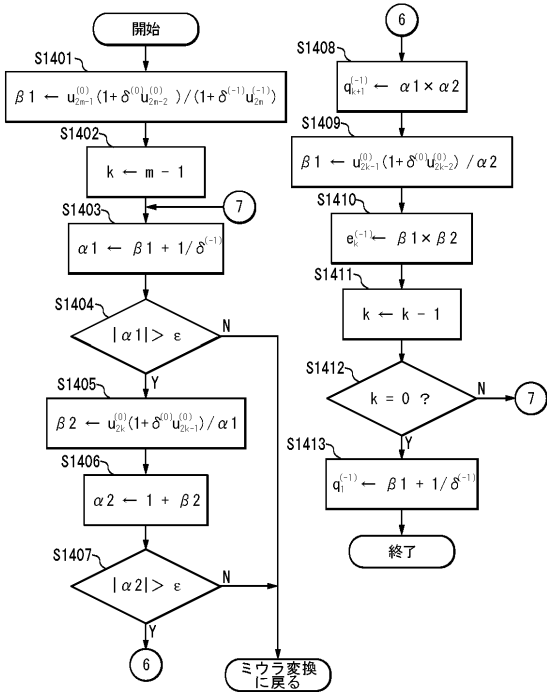
【図19】



【図20】

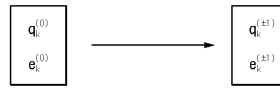


【図 2 1】

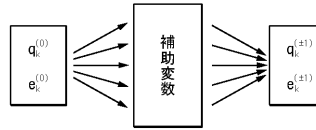


【図 2 2】

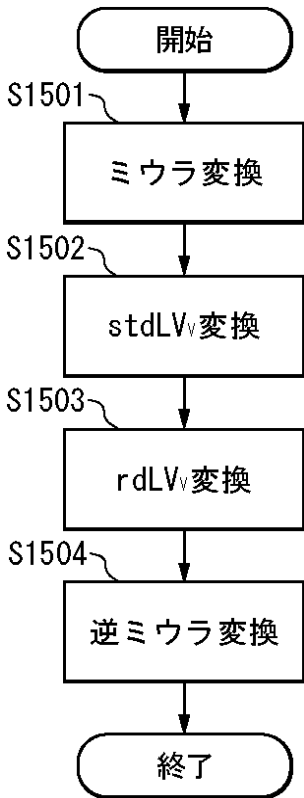
(a) qd型ツイスト分解法



(b) LV型ツイスト分解法

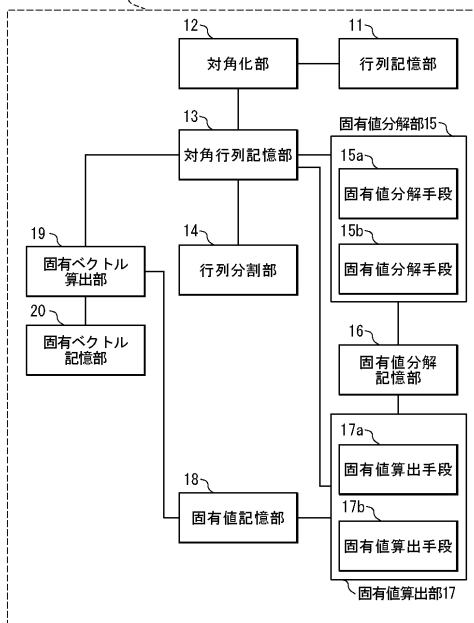


【図 2 3】

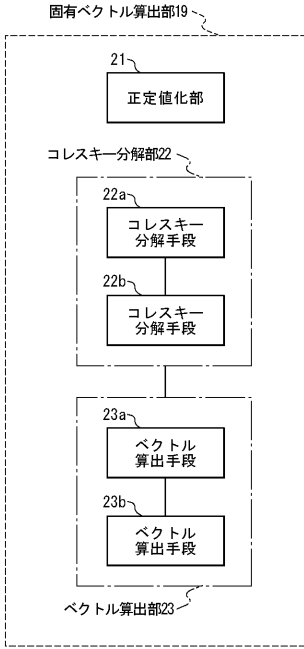


【図 2 4】

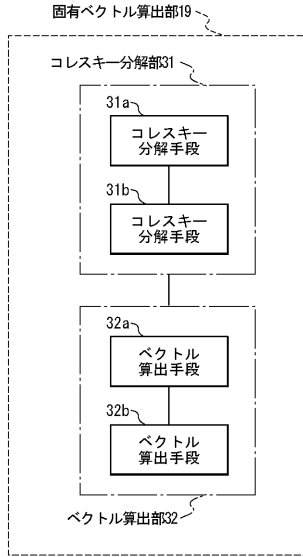
固有値分解装置1



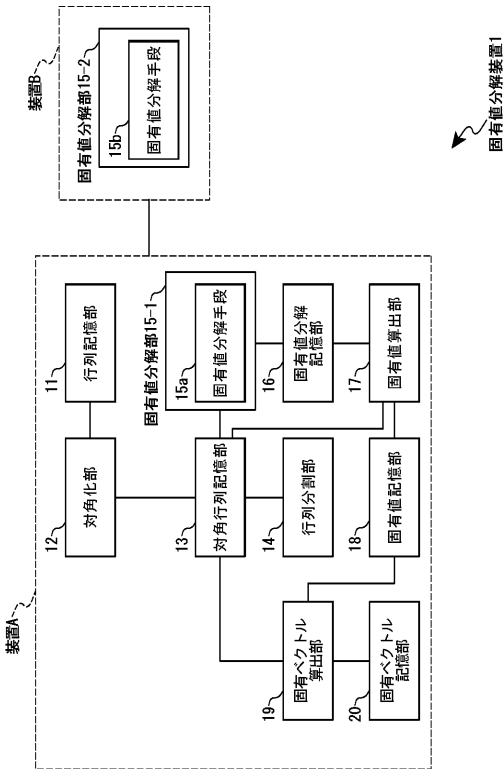
【図 25】



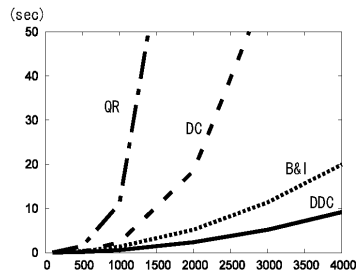
【図 26】



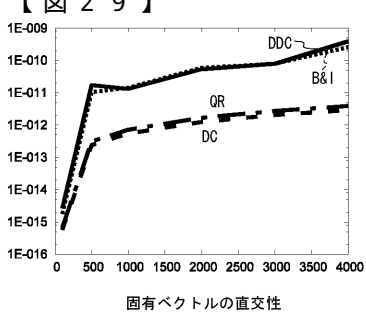
【図 27】



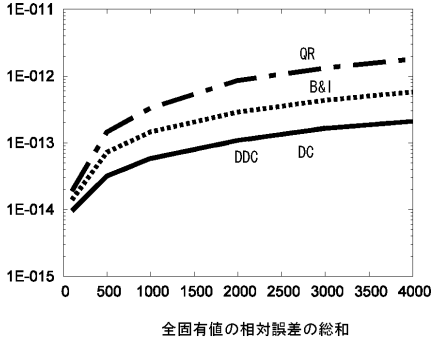
【図 28】



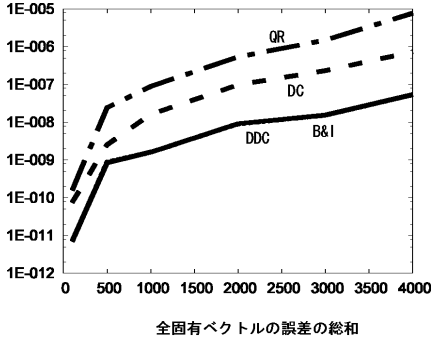
【図 29】



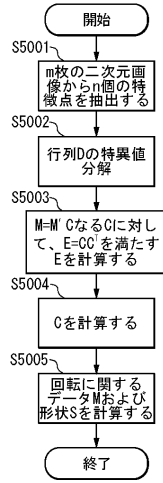
【図30】



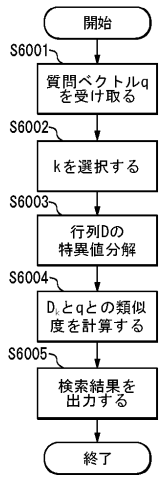
【図31】



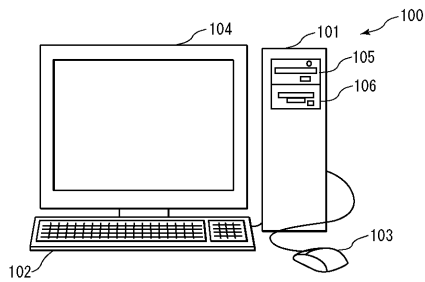
【図32】



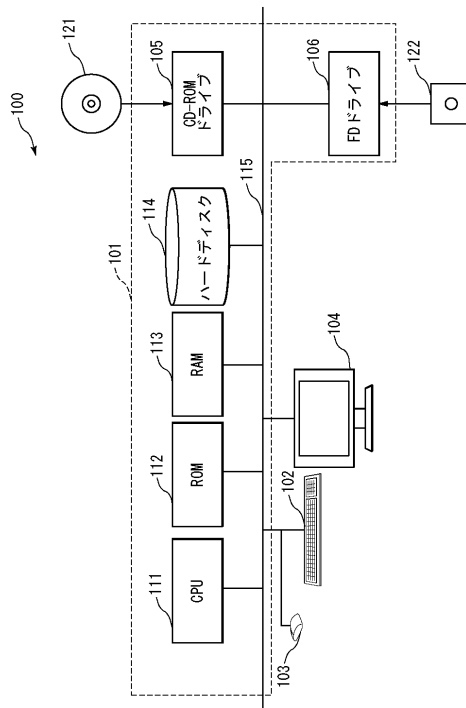
【図33】



【図34】



【図35】



## 【 国際調査報告 】

INTERNATIONAL SEARCH REPORT		International application No. PCT/JP2007/051575
<b>A. CLASSIFICATION OF SUBJECT MATTER</b> G06F17/13 (2006.01) i  According to International Patent Classification (IPC) or to both national classification and IPC		
<b>B. FIELDS SEARCHED</b> Minimum documentation searched (classification system followed by classification symbols) G06F17/13, G06F17/16  Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched Jitsuyo Shinan Koho 1922-1996 Jitsuyo Shinan Toroku Koho 1996-2007 Kokai Jitsuyo Shinan Koho 1971-2007 Toroku Jitsuyo Shinan Koho 1994-2007  Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)		
<b>C. DOCUMENTS CONSIDERED TO BE RELEVANT</b>		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Takahiro KATAGIRI, "Heiretsu Koyu Vector Keisan ni okeru Kyosei Taikaku Block-ka no Koka", Effect on Endiagonal-blocked Method for Parallel Eigenvector Computations, Information Processing Society of Japan Kenkyu Hokoku, Vol.2002, No.8 (2002-HPC-91) IPSJ SIG Notes, Information Processing Society of Japan, 31 August, 2002 (31.08.02), pages: 43 to 48, [CSDB] CS-NG-2004-00099-008	1-16
<input checked="" type="checkbox"/> Further documents are listed in the continuation of Box C. <input type="checkbox"/> See patent family annex.		
* Special categories of cited documents: "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier application or patent but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "&" document member of the same patent family		
Date of the actual completion of the international search 27 April, 2007 (27.04.07)		Date of mailing of the international search report 15 May, 2007 (15.05.07)
Name and mailing address of the ISA/ Japanese Patent Office		Authorized officer
Facsimile No.		Telephone No.

**INTERNATIONAL SEARCH REPORT**

International application No.

PCT/JP2007/051575

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Yusaku YAMAMOTO, Mitsuyoshi IGAI, Ken NAONO, "Hitaisho Sanju Taikaku Gyoretsu Muke no Heiretsu Renritsu Ichiji Hoteishiki Kaiho", A Parallel Linear Equation Solver for Nonsymmetric Tridiagonal Matrices, Transactions of Information Processing Society of Japan, Vol.43, No.SIG9 (HPS3) IPSJ, Information Processing Society of Japan, 15 August, 2001 (15.08.01), pages: 19 to 27, [CSDB] CS-NG-2003- 00274-002	1-16



国際調査報告		国際出願番号 PCT/JP2007/051575									
A. 発明の属する分野の分類 (国際特許分類 (IPC)) Int.Cl. G06F17/13(2006,01)i											
B. 調査を行った分野 調査を行った最小限資料 (国際特許分類 (IPC)) Int.Cl. G06F17/13 G06F17/16											
最小限資料以外の資料で調査を行った分野に含まれるもの <table border="0"> <tr> <td>日本国実用新案公報</td> <td>1922-1996年</td> </tr> <tr> <td>日本国公開実用新案公報</td> <td>1971-2007年</td> </tr> <tr> <td>日本国実用新案登録公報</td> <td>1996-2007年</td> </tr> <tr> <td>日本国登録実用新案公報</td> <td>1994-2007年</td> </tr> </table>				日本国実用新案公報	1922-1996年	日本国公開実用新案公報	1971-2007年	日本国実用新案登録公報	1996-2007年	日本国登録実用新案公報	1994-2007年
日本国実用新案公報	1922-1996年										
日本国公開実用新案公報	1971-2007年										
日本国実用新案登録公報	1996-2007年										
日本国登録実用新案公報	1994-2007年										
国際調査で使用した電子データベース (データベースの名称、調査に使用した用語)											
C. 関連すると認められる文献											
引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求の範囲の番号									
A	片桐 孝洋 TAKAHIRO KATAGIRI 並列固有ベクトル計算における強制対角ブロック化の効果 Effect on Endiagonal-blocked Method for Parallel Eigenvector Computations 情報処理学会研究報告 Vol.2002, No.80 (2002-HPC-91) IPSJ SIG Notes 社団法人情報処理学会 Information Processing Society of Japan 2002.08.31 pages: 43-48 [CSDB] CS-NG-2004-00099-008	1-16									
<input checked="" type="checkbox"/> C欄の続きにも文献が列挙されている。		<input type="checkbox"/> パテントファミリーに関する別紙を参照。									
* 引用文献のカテゴリー 「A」特に関連のある文献ではなく、一般的技術水準を示すもの 「E」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの 「L」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献 (理由を付す) 「O」口頭による開示、使用、展示等に言及する文献 「P」国際出願日前で、かつ優先権の主張の基礎となる出願		の日の後に公表された文献 「T」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの 「X」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの 「Y」特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの 「&」同一パテントファミリー文献									
国際調査を完了した日 27.04.2007		国際調査報告の発送日 15.05.2007									
国際調査機関の名称及びあて先 日本国特許庁 (ISA/JP) 郵便番号100-8915 東京都千代田区霞が関三丁目4番3号		特許庁審査官 (権限のある職員) 鳥居 稔	5B 8841								
		電話番号 03-3581-1101	内線 3545								

国際調査報告		国際出願番号 PCT/JP2007/051575
C (続き) . 関連すると認められる文献		
引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求の範囲の番号
A	<p>山本 有作 YUSAKU YAMAMOTO,            猪貝 光祥 MITSUYOSHI IGAI ,            直野 健 KEN NAONO ,            非対称三重対角行列向けの並列連立一次方程式解法            A Parallel Linear Equation Solver for Nonsymmetric Tridiagonal Matrices            情報処理学会論文誌 Vol. 43, No. SIG9 (HPS3) IPSJ            社団法人情報処理学会 Information Processing Society of Japan            2001.08.15            pages: 19-27            [CSDB] CS-NG-2003-00274-002</p>	1-16

## フロントページの続き

(81)指定国 AP(BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), EA(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), EP(AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OA(BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG), AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW

## 特許法第30条第1項適用申請有り

(72)発明者 岩崎 雅史

京都府京都市左京区吉田本町 国立大学法人京都大学大学院情報学研究所内

(72)発明者 高田 雅美

京都府京都市左京区吉田本町 国立大学法人京都大学大学院情報学研究所内

Fターム(参考) 5B056 BB42

(注)この公表は、国際事務局(WIPO)により国際公開された公報を基に作成したものである。なおこの公表に係る日本語特許出願(日本語実用新案登録出願)の国際公開の効果は、特許法第184条の10第1項(実用新案法第48条の13第2項)により生ずるものであり、本掲載とは関係ありません。