特別研究報告書

手術シミュレータにおけるSMW公式を用いた疎行列計算の高速化

指導教員　　富田 眞治 教授

京都大学工学部情報学科

依藤 逸

平成19年2月9日

手術シミュレータにおけるSMW公式を用いた疎行列計算の高速化

依藤 逸

内容梗概

近年の外科分野では，小切開創による低侵襲手術やロボットを用いたマイクロ手術など，手術の高度化が進んでいる．一方，開腹手術の安全な執刀には，一般的に数百例以上の開腹手術を経験しなければならない．しかし，短時間に経験できる手術数は限られており，技術の高度化と医師の技術向上がつりあっていないのが現状である．

そこで，実際の手術に代わって，手術シミュレータを用いた医療技術の習得訓練を行うことが望まれている．実際の手術を疑似体験できるようなインタラクティブで精巧なシミュレーションを行うことができれば，医師は技術を効率的に向上させることができると見込まれる．現在の手術シミュレータにおける計算は触診などの小変形の挙動表現に対してはインタラクティブに行うことができる．しかし，切断などの変形に対しては計算が追いつかず，挙動表現の実時間性は低い．

本稿ではSMW公式 (Sherman-Morrison-Woodbury foumula) を用いることで手術シミュレータにおける力-変位計算を高速に行い，切断を伴う変形に対してもインタラクティブな計算を行うことを目標とする．

従来の手術シミュレータ向けのライブラリでは，有限要素モデルにおける剛性マトリクスを用いた力-変位計算に対して変形を小変形に限定し，さらに線形性および固定境界条件が変化しないという制約のもとで計算を行う手法が用いられてきた．このような場合，剛性マトリクスが変形の状態に依存することはない．あらかじめ全剛性マトリクスの逆行列を1回求めておき，それを用いて直接法で変形ごとの力-変位計算を行う．逆行列を求める必要がなく，計算量は比較的少なくなるため，高速に計算することができる．

しかしながら，この手法を切断を伴う変形に対する力-変位計算に利用することはできない．これは，全体剛性マトリクスは変形の状態に依存し，あらかじめ求めておいた逆行列を用いる直接法での計算を行うことができないためである．また，切断を伴う変形が発生した時点で逆行列を再計算してその上で力-変

位計算を行った場合，変形ごとに逆行列を求める計算が必要とされる．しかし，直接法による逆行列の導出を行う場合には$O(n^3)$の計算量が必要とされるので，シミュレーションの実時間性を維持できるような高速な計算を行うことはできない．

そこで，切断を伴う変形に対しても実時間性を維持すべく，式変形を用いた逆行列の導出を考える．ここでは，微小時間後の剛性マトリクスに対する式変形を行う．これにより，微小時間後の逆行列が高速に導出できれば，切断を伴う変形が起きた場合においても実時間性を失わずに計算することが可能である．

式変形を行う際に，SMW 公式を用いて微小時間後の剛性マトリクスを導出を考えると，$n >> s$であるsを用いて$O(s^3)$で微小時間後の逆行列が導出できる．SMW 公式の実装では，転置行列の作成による配列の局所性の利用，疎行列性の利用，並列化などを用いて最適化を行う．

今回扱う，SMW 公式で行う計算は，実際の大動脈の剛性マトリクス aorta からの微小時間後の逆行列の導出である．大動脈のモデルには，ノード数 197，388，799，1661 の 4 つもモデルを用いる．比較のために，反復法である共役勾配法を用いて解を導出するのに必要とされた時間も調査した．

行列の成分変化をさまざまな割合で行った場合における単一の計算機での実行結果は,「SMW 公式に加えて転置行列の作成と CRS 形式と CCS 形式による疎行列性の利用」を行ったプログラムが全てのノード数において最も高速であった．

並列化したプログラムでも同様の評価を行ったが，今回は手術シミュレータにおける剛性マトリクスの微小時間後の導出を行うとき，行列の成分が全要素の 10 %変化すると仮定した．ノード数が小さいモデルにおいては並列化による速度改善は見られなかったが，ノード数が大きくなるに従って並列化による効果は顕著になった．aorta388 以下のノード数の小さいモデルにおいては単一の計算機で計算を行ったほうが高速であり，aorta799 以上のノード数の大きいモデルではクラスタを用いて並列化を行う手法が有効であるということがわかった．

High-Speed Calculation Method with SMW Formula for Sparse Matrix on a Surgical Simulator

Suguru YORIFUJI

Abstract

Recently, in the field of surgery, the surgery with a finesse incision and a microsurgery using a robot is the center of attention because of high technologies for those surgeries. On the other hand, a doctor must have experienced more than 100 abdominal operations to get used to such kind of surgery. Actually, it is difficult for doctors to have such a large number of abdominal operations in a short period of time so that they cannot improve their surgical skills easily.

There's needs that doctors improve their surgical skills by using surgical simulators. If doctors can practice surgery as if they really operate for patients with surgical simulators, we can expect that they can improve their surgical skills effectively. Currently, the surgical simulators can interactively perform calculations for behavioral expression under small changes of an object such as manipulation, but the surgical simulators can less interactively perform calculations for behavioral expression of changes caused by cutting.

In this paper, I challenged to improve calculation speed with SMW formula (Sherman-Morrison-Woodbury formula) to achieve interactive calculation under changes changes caused by cutting.

A conventional Real Time Physics-Based Simulation Library for the surgical simulator has used Hirota method to perform a calculation with a Stiffness Matrix of finite element model. Hirota method limits transformation to small one and perform calculations under the restriction that linearity and the conditions for fixed boundary does not change.

Under this restriction, a Stiffness Matrix does not depend on the condition of transformation. Therefore, if once we find the inverse matrix of a stiffness Matrix, a calculation for expression of displacement by stress is performed with direct method, using the inverse matrix found before. So, we do not have to find the new inverse matrix and we can perform fast calculation with small calculation amount. In this way, under this restriction, a calculation for expression of displacement by stress is performed fast by Hirota method.

However, a calculation for behavior expression under changes with cutting cannot be performed by Hirota method, because change with cutting makes the restriction meaningless. When the stiffness matrix is changed, the calculations cannot be performed with direct method, using the inverse matrix which is found previously. If the calculation under changes with cutting is performed by direct method which use the inverse matrix, we have to calculate the inverse matrix in every transformation. To find the inverse matrix, we have to calculate $O(n^3)$ calculation. It is too heavy calculation so that we cannot maintain the interactivity of the simulation.

To maintain the interativity, I propose the method which use transeformation of formula to find the inverse matrix. By using stiffness matrix after a moment which is derived by appling SMW formula to the old inverse matrix,we can express the new inverse matrix. It only needs small acount of calculation $O(s^3)(n >> s)$. Furthermore, I applied various optimization when I implemented SMW formula.

I evaluated proposed method with a calculation of the new inverse matrix of aorta, after a moment. The nodes of aorta are 197, 388, 799, and 1661, respectively. For comparison, I evaluated execution time of Conjugate Gradient method which is one of repetition method.

Firstly, I evaluated proposed method under changing few components of matrix and single computer. The result shows that the method "using SMW formula with creating transpose matrix and utilizeing sparse matrix under CRS format and CSS format" is most rapid on all aorta.

I evaluated the excution time using parallel computer on similar situation. In this case, when I found the new inverse matrix, I supposed that 10 percents of all components has changed. If the number of nodes is small, parallel computer does not work. But, if the number of nodes is large, parallel computer works well. The results show that if the number of nodes is less than 388, using single computer is more rapid than parallel computer. On the other hand, if the number of nodes is more than 799, using parallel computer is more rapid than using single computer.

手術シミュレータにおけるSMW公式を用いた疎行列計算の高速化

目次

第1章　はじめに

近年，医療技術は急速な進化・発展を遂げている．だが，それに伴い技術難度も上昇し，特に外科分野では，新しい手術手技に対して外科医のスキルを如何に効率的に向上させ，一定の水準に到達させるかが大きな課題となっている．この課題に対し，人体組織の力学や生理をモデル化した仮想物体を用いて手術シミュレーションを実現し，スキル向上に役立てるアプローチが考えられる．我々は仮想的な空間において手術シミュレーションを容易に行える，インタラクティブな手術シミュレータの実現を，京都大学医学部附属病院医療情報部と共同で目指している [1][2][3].

手術シミュレータにおいて，どのようなことが必要とされているのかについて考えてみる．まず第一に必要とされるのは，実時間性である．実際に精巧にシミュレーションを行うことが可能であったとしても，大幅に遅れたリアクションであっては無意味である．リアルタイムで挙動に対する反応が必要とされる手術シミュレータの変形計算において，実時間性を維持するためには 30Hz のリフレッシュレートを要する [4]．医師の技術習得の助けとなる必要があるので，実際の医師が行う挙動と同じことをシミュレーションできる必要がある．そのことを考えると，医師が触診を行った際に実際の人体と同じような反応が返ってくるということが要求される．また，実際の手術においては人体切開を行うことが必要である．そのため，切離あるいは剥離にも対応することができるようなモデルであるということも手術シミュレータに要求される性能である．

臓器の触診等で加わる力とその力による変位の表現には，有限要素モデルを用いるのが一般的である．中尾らが開発した，手術シミュレータ向け実時間力学計算手法のライブラリ [4] における有限要素モデルに基づく反力，変形計算手法では，広田らが提案した近似解法 [5](以下，広田手法と呼ぶ) が用いられている．有限要素モデルにおいて，線形性と固定境界条件が変わらないという制約のもとで全体剛性マトリクスは変形の状態に依存しない．このことから，逆行列をあらかじめ求めておくことで，直接法で力-変位計算を容易かつ高速に行うことを可能にする手法である．線形性と固定境界条件が変わらないという制約は臓器の触診といった小変形では適用可能である．小変形については 2.2.1 節で述べる．

切開手術ではシミュレーション対象物体の幾何学的な構造 (トポロジ) 自体が

変化するため，切開を伴う変形が発生した場合には剛性マトリクス自体が変化してしまう．従って広田手法が仮定している線形性が成立しなくなる．そのため，小変形のみを仮定してのシミュレーションにおいて，切断手技をリアルタイムで表現することは現在の技術では難しい．

そこで，実時間応答性を改善するということが本研究の目的である．

第2章　研究背景

切断手技は患部治療に必要不可欠だが，失敗すれば患者の生命に関わることもある重要な手技である．よって，切断手技の挙動をリアルタイムで忠実に表現できることは医師の切断技術の向上へとつながり，結果としてスムーズで安全な手術を行うことにもつながる．つまり，手術シミュレータの完成には欠かせない要素と言える．

リアルな挙動を表現するため，臓器のモデル化には有限要素法を用いる [4]．有限要素法における力-変位計算について，従来の計算手法では小変形のみを仮定している．小変形のみであるという仮定の下では剛性マトリクスは変化せず，あらかじめ逆行列を求めておけば力から変位の計算は直接法により高速に行えた．しかし，仮定に該当しない切断を伴う変形では剛性マトリクスは変化し，新たな剛性方程式を解く，ひいては直接法で力-変位計算を行うためには新たな逆行列の導出が必要となる．剛性方程式を解くために直接法を用いて逆行列の導出を行う際に必要とされる時間は，$O(n^3)$ である．マトリクスの次数が大きくなってくると，毎回逆行列を求めていては，実時間性が保証できない．この問題を解決するために，直接法を用いて解を得るのではなく，$O(n^2)$ で解を求めることが可能な共役勾配法を用いることでリアルタイムでの表現を達成しようとしたが，現時点ではまだ十分な高速化ができたとはいえない．一般に大規模な疎行列問題に対してはメモリ容量の制限から反復法を用いるのが常套手段ではあるが，現時点での本研究の対象としている行列のサイズではかろうじて直接法でも解くことが可能な範囲である．そこで，本研究では行列 K とその逆行列 K^{-1} が与えられているときに，K とわずかに異なる行列 K' の逆行列を高速に求める手法を用いて，手術シミュレーションの高速化の可能性を検証する．

本章では，有限要素モデルの概要，剛性マトリクスの変化の有無，SMW 公式について述べる．

2.1 有限要素モデル

有限要素モデルとは，変形に対し無限の自由度を持つ物体を有限の自由度を持つ要素（有限要素）の集合体と近似し，この集合体に加わる力とその変位の関係に対して成立する方程式（連立一次方程式）を解くモデルである [6]．全自由度に加わる力のベクトルを$\boldsymbol{F}$，全自由度の変位ベクトルを$\boldsymbol{u}$とする．係数行列を K とすると，この連立一次方程式は，

$$\boldsymbol{F} = K\boldsymbol{u} \tag{1}$$

と表される．ここで K は，物体に加わる力とその変位の関係を表すマトリクスとして，全体剛性マトリクスと呼ばれる．全体剛性マトリクスは，要素ごとの要素内全自由度の力-変位関係を表現する要素剛性マトリクスを，要素の集合全体について足し合わせることで得られる．モデルが持つすべての自由度に対して変位が与えられた際に，各自由度に発生する力を与えるものである．ここで線形なモデルに限って考えると，全体剛性マトリクスは定数行列となる．すると，全体剛性マトリクスおよび全自由度の力-変位関係は次のように与えられる．

$$\begin{bmatrix} \boldsymbol{f}_i \\ \boldsymbol{f}_o \end{bmatrix} = \begin{bmatrix} K_{ii} & K_{io} \\ K_{oi} & K_{oo} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_i \\ \boldsymbol{u}_o \end{bmatrix} \tag{2}$$

ここで，$\boldsymbol{u}_i$ および $\boldsymbol{f}_i$ は力が与えられた接触点のノードにおける変位および力，$\boldsymbol{u}_o$ および $\boldsymbol{f}_o$ は接触点以外の点のノードにおける変位および力である．要素（メッシュ）として四面体を用い，各節点（ノード）における自由度は x，y，z の 3 つである．図 1 に物体のメッシュ分割のイメージを示す．

2.2 剛性マトリクスの小変形と大変形

2.2.1 剛性マトリクスを変化させない場合

ある時間 t_0 での剛性マトリクス K_0 と，微小時間 δt 経過後の時刻 t_1 における剛性マトリクス K_1 は，厳密には変化してはいるが，多くの成分が等しいはずである．そこで，時刻 t_0 と時刻 t_1 における剛性マトリクスが等しいと考えて，剛性マトリクスを変化させないで力を加えて行う仮想物体の変形を小変形と呼ぶ．小という言葉を冠してはいるが，物体が物理的に大きな変形をしていたとしても，剛性マトリクスが変化していないのであれば，便宜上それを小変形と呼ぶ．

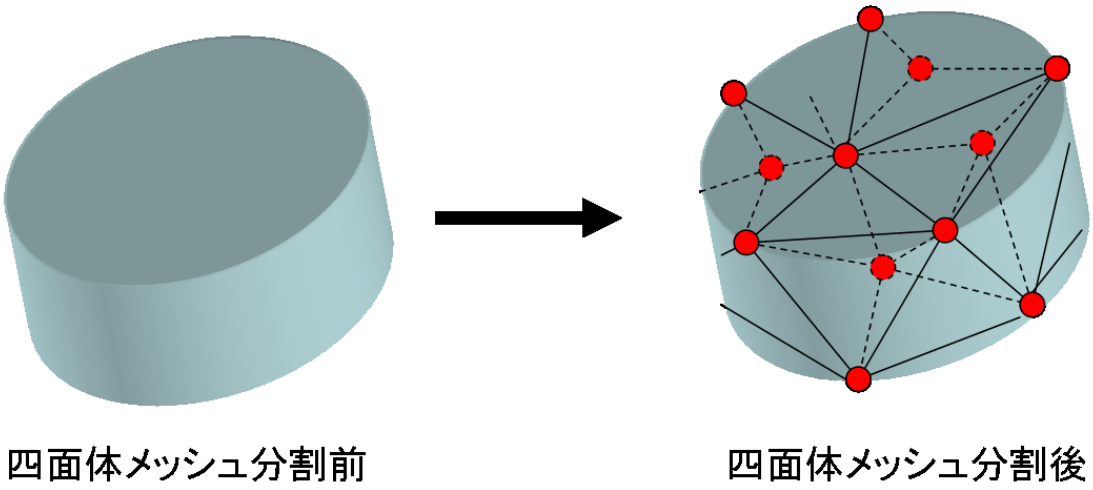

図 1: 要素 (メッシュ) 分割

2.2.2 剛性マトリクスを変化させる場合

ある時間 t_0 での剛性マトリクス K_0 と，そこから微小時間 δt 経過後の時刻 t_1 における剛性マトリクス K_1 は，ほとんど等しいはずではあるが厳密に考えると等しくはない．そこで，剛性マトリクスを変化させながら力を加えて行う仮想物体の変形を大変形と呼ぶ．実際の人体に可能な限り忠実にシミュレーションを行おうとするのであれば，小変形のみでは不可能であり，大変形を組み合わせねばならない．大という言葉を冠してはいるが，物体が物理的に大きく変形していなかったとしても，剛性マトリクスが変化していたとするのであればそれは大変形として扱われる．本節で述べた小変形，大変形は変形の度合を表しているというわけではなく，剛性マトリクスを基準とした変形の種類を表しているのである．

2.3 SMW 公式

SMW 公式とは，Sherman-Morrison-Woodbury formula の略称であり，以下のような式で表される [7]．ただし，ここでの A, B, C, D はそれぞれ行列であり，なおかつ A, D は逆行列が存在するという条件がある．

$$(A + BDC)^{-1} = A^{-1} - A^{-1}B(D^{-1} + CA^{-1}B)^{-1}CA^{-1} \tag{3}$$

2.3.1 式の導出

式の導出は以下のようにして与えられる．ある行列 X が正則であるとすれば，その逆行列 X^{-1} が存在して，

$$XX^{-1} = I \tag{4}$$

が成立する．ここで，行列 X を次のように分割した式を考えてみる (このとき，M_{11} および M_{22} は正方行列であるとする)．

$$\begin{bmatrix} A & B \\ C & D^{-1} \end{bmatrix} \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \tag{5}$$

式 (5) を計算して以下のような式に展開する．

$$AM_{11} + BM_{21} = I \tag{6}$$

$$AM_{12} + BM_{22} = 0 \tag{7}$$

$$CM_{11} + D^{-1}M_{21} = 0 \tag{8}$$

$$CM_{21} + D^{-1}M_{22} = I \tag{9}$$

ここで，式 (8) の両辺に左から BD を乗じると，

$$BDCM_{11} - BM_{21} = 0 \tag{10}$$

となる．式 (6) と式 (10) の和をとって整理すると，

$$M_{11} = (A + BDC)^{-1} \tag{11}$$

が得られる．さらに，式 (6) の両辺に左から CA^{-1} を乗じて，

$$CM_{11} + CA^{-1}BM_{21} = CA^{-1} \tag{12}$$

式 (12) から式 (8) を減じて整理すると，

$$M_{21} = (D^{-1} + CA^{-1}B) \tag{13}$$

式 (13) を式 (6) に代入して整理すると，

$$AM_{11} = I - B(D^{-1} + CA^{-1}B)^{-1}CA^{-1} \tag{14}$$

A には逆行列が存在するという仮定があるので，式 (13) に左側から A^{-1} を乗じて式 (15) を導く．

$$M_{11} = A^{-1} - A^{-1}B(D^{-1} + CA^{-1}B)^{-1}CA^{-1} \tag{15}$$

式 (11) と式 (15) から，式 (3) が導かれる．

反復法を用いた解の導出には $O(n^2)$ かかるのに対して，SMW 公式を用いる場合に必要とされる逆行列を求める時間は，行列 $(D^{-1}+CA^{-1}B)$ が大きさ $s \times s$ の行列だとすれば，直接法を用いても $O(s^3)$ である．微小時間経過ということをふまえると，通常 $n >> s$ であるので，単純に直接法を用いて逆行列を求める場合と比較すると，SMW 公式を用いた場合の逆行列導出時間は短縮されることが期待できる．

表 1 にて計算量およびメモリ使用量の比較を行う．共役勾配法の計算量とメモリ使用量は $O(n^2)$ であるが，n^2 の係数はそれぞれ反復回数 l と 1 行あたりの平均非ゼロ要素数 k に比例する．

表 1: 計算量およびメモリ使用量の比較

	計算量	メモリ使用量
直接法 (LU 分解)	$O(n^3)$	$O(n^2)$
共役勾配法	$O(n^2)$	$O(n^2)$
SMW 公式	$O(n^2)$	$O(n^2)$

第 3 章　手術シミュレータで使用する計算モデル

3.1　本研究の対象とする手術シミュレータの現在の計算モデル

考案されている医療シミュレータの計算手法について以下に述べる．発生する操作を小変形に限定したシミュレーションにおいて，接触点における力が分かっていて，接触点以外では力は加わらないと仮定する．有限要素モデルにおける各自由度の変位の導出は式 (16) から式 (19) のようになる．まず，式（2）

より

$$\boldsymbol{f}_i = K_{ii}\boldsymbol{u}_i + K_{io}\boldsymbol{u}_o \tag{16}$$

$$\boldsymbol{f}_o = K_{oi}\boldsymbol{u}_i + K_{oo}\boldsymbol{u}_o \tag{17}$$

である．式（17）にて，仮定より $\boldsymbol{f}_o = \boldsymbol{0}$ であるから，

$$\boldsymbol{u}_o = -K_{oo}^{-1}K_{oi}\boldsymbol{u}_i \tag{18}$$

式（16）へ代入して，

$$\boldsymbol{f}_i = K_{ii}\boldsymbol{u}_i - K_{io}K_{oo}^{-1}K_{oi}\boldsymbol{u}_i \tag{19}$$

となる．

マトリクスが数千次となる場合，式 (18)，(19) で逆行列を求める計算は多大な計算時間を要する．触診等の小変形においては，変形前後で剛性マトリクスの変化は非常に小さく，近似により剛性マトリクスは同一のものと見なすことができる．よって，一連の操作によって小変形しか起こらないと仮定した場合，式 (18)，(19) の逆行列を一度求めておけば各時点での力-変位計算はその逆行列を用いることができ，直接法により容易に計算できる [5]．しかし，切断を伴う変形が起こると変形の前後で剛性マトリクスが大幅に変化し，変形前の逆行列を利用できない．その結果，変化ごとに新たな逆行列を導出する必要が生じ，その導出に多大な時間を要する．

現在のところ，中尾らはある大きな変化が起きる時点で剛性マトリクスを更新するという手法を採用している．この手法を用いる具体例として，ある物体をメスで切開するという操作を行うことを考える．その操作における時系列は，「切開前」,「切開中」,「切開後」の 3 つの状態を考えることができる．切開前，切開中，切開後である時間をそれぞれ t_{before}，t_{during}，t_{after} とする．t_{before}，t_{after} の間は，それぞれ切開前，切開後の剛性マトリクスを用いてシミュレーションを行う．t_{during} においては，切開前，切開後のマトリクスの連続で切開操作を表現する．切開前から切開後へのモデルの遷移が小変形におさまる範囲で発生するのであれば，大変形を起こす前に近似的に切開動作を表現するということが可能になる．

3.2　目標とする手術シミュレータの計算モデル

ここで，時刻 t_0〜t_n までそれぞれの時間に対応する剛性マトリクスが存在しているとき，時間 t_i に対応する剛性マトリクスの逆行列 ${K_i}^{-1}$ をどのようにして求めるのかということについて考えてみる．求める手法には，

1. 直接法を用いて A_i から逆行列を求める
2. 一つ前の時刻での逆行列 ${A_{i-1}}^{-1}$ に対して SMW 公式を適応することで逆行列を導出する
3. いくつか前の時刻での逆行列 ${A_{i-k}}^{-1}$ に対して SMW 公式を適応することで逆行列を導出する

の 3 つが考えられる．2 の手法を用いると，最も計算量は少なくなるが，誤差が蓄積していってしまうことが問題である．3 の手法を用いると，2 の手法と比較して誤差は $1/k$ になるが，2 の手法に比べて計算に必要な時間が多くなってしまう．1 の手法では誤差が出ることはないが，必要とされる計算時間は最も多くなってしまう．これらの手法をどのように使うのかということで計算精度および計算速度は大きく変わってきてしまう．剛性マトリクスから力-変位計算を行う，つまり剛性マトリクスを係数行列に持つ連立一次方程式を解く方法として，直接法と反復法が考えられる．直接法は逆行列を用いる行列演算に帰着される．広田手法の仮定のもとでは、あらかじめ導出しておいた逆行列を用いれば新たな逆行列の導出は不必要であった．しかし、大変形のもとでは剛性マトリクスの変化は大きく、新たな剛性マトリクスに対しての逆行列の導出が必要とされる．一般に直接法を繰り返し用いて逆行列を求めようとした場合に必要とされる計算時間は $O(n^3)$ である。野田らが行った、反復法を用いた場合の計算量は $O(n^2)$ である．

野田らは反復法の一つである共役勾配法を用いて医療シミュレータ計算を行ったが，十分な速度を得ることはできなかった．そこで，シミュレーションを始める前の状態での剛性マトリクスに対する逆行列が既知であるという仮定のもとに、SMW 公式を適用することで逆行列の導出を行うことにする．SMW 公式において逆行列を求める場合，連続的な時系列での適用ということを考えると、時系列間の差分行列は疎行列となるので，逆行列を求める時間は大きく短縮されることが期待できる．

3.3 手術シミュレータへのSMW公式の導入

手術シミュレータにおいて30Hzのリフレッシュレートを達成するためには，単純計算で1秒間に30回ほど式(1)を解く必要がある．ここで，一度解を求めるたびに式(1)において$\boldsymbol{F}$とKが変化するとすれば，ひとつ前の時系列を考えると，解を求めたときからの変化分δF，δKをそれぞれ用いて，

$$\boldsymbol{F'} = \boldsymbol{F} + \delta\boldsymbol{F} \tag{20}$$

$$K' = K + \delta K \tag{21}$$

というように，新たに置き直すことができる．また，このとき求めるべき解を$\boldsymbol{u'}$と置くと，次のように表すことができる．

$$\boldsymbol{u'} = K'^{-1}\boldsymbol{F'} \tag{22}$$

式(21)では，経過しているのは30分の1秒という微小時間であるので，K'とKはほとんど変化していない．つまり，変化分δKという行列では多くの成分が0となっている．

δKの表現について考えてみる．大きさ$n \times n$の正方行列δKから全ての成分が0となっている列を取り除いた行列をδK_cとする．全ての成分が0となっている列の数が$n - s$であるとすれば，δK_cの大きさは$n \times s$となっている．このとき，成分が0と1からのみから構成される大きさ$s \times n$の行列E_cを用いることで，式(23)のように書くことができる．

$$\delta K = \delta K_c \times E_c \tag{23}$$

式(23)のような分解が実際に存在することを以下に示す．δK_cは大きさ$n \times s$の行列であるので，式(24)のように書くことができる．

$$\delta K_c = \begin{pmatrix} k_{11} & \cdots & k_{1s} \\ \vdots & \ddots & \vdots \\ k_{n1} & \cdots & k_{ns} \end{pmatrix} \tag{24}$$

0と1で構成される行列E_cを考えるが，ここで式(25)のように書くことができ

る大きさが $s \times s$ である単位行列 E を考える.

$$E = \begin{pmatrix} 1 & 0 & & & \mathbf{0} \\ 0 & 1 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & 1 & 0 \\ \mathbf{0} & & & 0 & 1 \end{pmatrix} \tag{25}$$

δK_c は$\boldsymbol{k_i}$= $\begin{pmatrix} k_{i1} & k_{i2} & \cdots & k_{is} \end{pmatrix}$ である行ベクトル$\boldsymbol{k_i}$を用いて式 (26) のように書くことができる.

$$\delta K_c = \begin{pmatrix} \boldsymbol{k_1} \\ \boldsymbol{k_2} \\ \vdots \\ \boldsymbol{k_n} \end{pmatrix} \tag{26}$$

式 (25) は，i 番目の要素のみが 1 で残りの要素が 0 である列ベクトル$\boldsymbol{e_i}$を用いて式 (27) のように書くことができる.

$$E = \begin{pmatrix} \boldsymbol{e_1} & \boldsymbol{e_2} & \cdots & \boldsymbol{e_s} \end{pmatrix} \tag{27}$$

ここで，行列 E の m 列目に全ての要素が 0 の列を挿入した式 (28) のような行列 E_c を考える.

$$E_c = \begin{pmatrix} \boldsymbol{e_1} & \cdots & \boldsymbol{e_{m-1}} & \boldsymbol{0} & \boldsymbol{e_m} & \cdots & \boldsymbol{e_s} \end{pmatrix} \tag{28}$$

$A = \delta K_c \times E_c$ の演算を行うと，$j < m$ のときは,

$$a_{ij} = \boldsymbol{k_i} \times \boldsymbol{e_j} = k_{ij} \tag{29}$$

$j = m$ のときは,

$$a_{ij} = \boldsymbol{k_i} \times \boldsymbol{0} = 0 \tag{30}$$

$j > m$ のときは,

$$a_{ij+1} = \boldsymbol{k_i} \times \boldsymbol{e_j} = k_{ij} \tag{31}$$

よって，式 (29) と式 (30) と式 (31) から，式 (32) が導かれる．

$$A = \delta K_c \times E_c = \begin{pmatrix} k_{11} & \cdots & k_{1j-1} & 0 & k_{1j} & \cdots & k_{1s} \\ \vdots & & \vdots & \vdots & \vdots & & \vdots \\ k_{n1} & \cdots & k_{nj-1} & 0 & k_{nj} & \cdots & k_{ns} \end{pmatrix} \tag{32}$$

式 (32) からわかる通り，A は δK_c の k 列目に全ての要素が 0 の列が挿入された大きさ $n \times (s+1)$ の行列となっている．よって，δK から δK_c を作成する際に x 列目を取り除いたとすれば，大きさ $s \times s$ の単位行列の x 列目に全ての要素が 0 となっている列を挿入すれば，$\delta K_c \times E_c = \delta K$ を満たす行列 E_c を作ることが可能である．この操作を再帰的に行うことで，複数の列を抜いた場合にも対応する E_c を作成することが可能である．よって，式 (23) のような分解が存在することが示された．

式 (23) を用いると，式 (21) は次のように書くことができる．

$$K + \delta K = K + \delta K_c \times E_c \tag{33}$$

式 (3) において，$A = K$,$B = \delta K_c$,$C = E_c$,$D = I$ と置換して整理すると，式 (34) が導かれる．

$$(K + \delta K_c E_c)^{-1} = K^{-1} - K^{-1}\delta K_c (I + E_c K^{-1} \delta K_c)^{-1} E_c K^{-1} \tag{34}$$

式 (20) と式 (21) と式 (34) から式 (22) を解くと，式 (35) が得られる．

$$\boldsymbol{u'} = (K^{-1} - K^{-1}\delta K_c (I + E_c K^{-1} \delta K_c)^{-1} E_c K^{-1})(F + \delta F) \tag{35}$$

最終的に式 (35) のような形で書いたが，次の時系列のために必ず K'^{-1} を求める必要があるので，実際には $(K^{-1} - K^{-1}\delta K_c (I + E_c K^{-1} \delta K_c)^{-1} E_c K^{-1})(F + \delta F)$ を展開して解くことはできず，式 (36) の形で解を求めることになる．

$$\boldsymbol{u'} = K'^{-1}(F + \delta F) \tag{36}$$

第 4 章　SMW 公式の実装と高速化

4.1　転置行列の作成

行列の乗算 $C = A \times B$ を行う場合に，式 (37) のような計算を行うことになる．

$$C[i*n+j] = A[i*n+0]*B[0*n+k] + A[i*n+1]*B[1*n+k] + \cdots + A[i*n+n-1]*B[(n-1)*n+k] \tag{37}$$

この行列演算を一次元配列を用いて実装するのだが，行列 B の要素を格納する配列 B に対してはポインタを n ずつとばしてアクセスしている．この状態では配列の局所性を全く利用することができていない．この問題を解消するために，乗算結果を転置して保存することができるようにした．行列 B が転置されていれば配列の局所性を利用することができるので，式 (34) を計算する際に後ろから計算を行うようにすれば，計算を高速化することができる．

4.2 行列のデータ構造の計算への特化

SMW 公式を用いる際に出現する δK_c，E_c は疎行列である場合が多い．例えば行列の乗算 $C = A \times B$ を行う際に，式 (37) のような演算を考える．ここで，行列 A，B の成分のうち多くがゼロ要素であった際に何の対策も講じなければ，式 (37) において，$0 + 0 + \cdots + x + 0 + \cdots$ のように，0 の和算という結果的に無意味な演算を行うことになってしまう．このような状態を避けるために，行列のデータ構造を疎行列の乗算に特化した形へと変えてやることを考える [8].

本研究で用いる代表的な疎行列を可視化して図 2 に載せる．黒い部分が非ゼロ要素，白い部分がゼロ要素である．

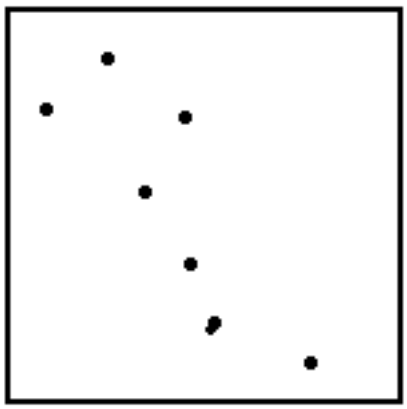

図 2: 疎行列の成分分布

4.2.1 CRS 形式

CRS 形式 (Compressed Row Storage format) では，行列を行方向から圧縮する．圧縮するためには，3 つの配列を持つ構造体を用いる．構造体は，

- 非ゼロ要素の値を記憶する配列 val

- 非ゼロ要素のインデックス記憶する配列 ind
- 非ゼロ要素の列番号を記憶する配列 ptr

から構成される．

4.2.2 CCS 形式

CCS 形式（Compressed Column Storage format）では，行列を列方向から圧縮する．圧縮するためには，3 つの配列を持つ構造体を用いる．構造体は，

- 非ゼロ要素の値を記憶する配列 val
- 非ゼロ要素のインデックス記憶する配列 ind
- 非ゼロ要素の列番号を記憶する配列 ptr

から構成される．

4.3 並列化

SMW 公式では行列演算を主に行うため，行列を分割して処理を並列化することによる処理速度の向上が見込まれる．並列化にあたっては PC クラスタを使用し，プログラミングにあたっては MPI を使用する．並列化部位は，計算全体の終盤にある密行列演算に対して適用する．

4.3.1 PC クラスタ

PC クラスタとは，複数の PC をネットワークで接続し，仮想的に一台の並列コンピュータとして利用することでパフォーマンスを向上させた PC 群を言う．PC クラスタは 2 台から 8 台の小規模なクラスタから，数千台規模でスーパーコンピュータの性能を発揮する PC クラスタを構成することが可能である．近年の PC の高性能化，低価格化に伴うコストパフォーマンスの向上により，PC クラスタの構成もコストパフォーマンスの向上が図られた．また，ネットワークにおいても 100M ethernet から Gigabit Ethernet や Myrinet などの高速ネットワークの登場により，PC の高性能化に対するバランスが取れるようになり，大規模な PC クラスタで発生するネットワークのオーバヘッドを最小限に抑えることが可能となった．なお，今回使用するクラスタは 8 台の小規模なクラスタである．

4.3.2 MPI

本研究室のクラスタは分散メモリ型である．よって，プログラミングにあたってはデータを通信する必要が生じる．この通信を実装するためにメッセージパッシング型のプログラミングである MPI(Message Passing Inteface) を使用する

[9]．MPI はメッセージ通信のプログラムを記述するために広く使われる標準化を目指して作られた，メッセージ通信の API 使用である．MPI は，ネットワーク上のプロセス間の効率の良い通信が可能であること，異なるプラットフォーム上での移植の容易性の保証，信頼できる通信インタフェースの提供，PVM など既存のものと同様の使いやすさと高い融通性を実現していることを特徴とする．

4.3.3　密行列演算の並列化

SMW 公式の速度を低下させる原因として，密行列演算の存在があげられる．有限要素モデルに出現する剛性方程式を解く際に，この密行列演算の次数は数千次に及ぶことも多い．密行列演算を並列に処理することができれば処理速度の向上が見込まれる．SMW 公式の計算に出現する密行列演算としては，

- 行列同士の加算
- 行列同士の減算
- 行列同士の乗算

が挙げられる．密行列演算を並列に行うためには行列を各ノードへ分配する必要がある．n 行 m 列 の行列を p ノードに分配する場合，1 ノードあたり $n/p*m$ 個の要素数を持つように分配する．ただし，n が p の倍数となっていない場合には，ある 1 ノードのみが $(n/p+l)*m$ 個の要素を持つものとする．ここで l は n の p に対する剰余である．分割を行ったあと，行列同士の乗算を並列に行うため，加算および減算はそのままワーカで計算することが可能であるが，行列同士の乗算を行う場合はワーカのみでは計算を続けることができない場合がある．このような場合は，ワーカからマスタにデータを送信してから再びマスタからデータを受け取る必要がある．

第 5 章　評価

5.1　実行環境

評価環境を表 2 に示す．使用するクラスタはマスタ 1 台にワーカ 8 台である．

5.2　実行結果

まず，SMW 公式を用いる場合に扱うべき問題としては，大変形が行われた際にリフレッシュレート 30Hz を維持することが可能なのかということである．そこで，剛性方程式を解く際に必要とされる逆行列の導出についての評価を行

表 2: 評価環境

ノード数	8 台
CPU	Pentium(R) 4 3.40 GHz
OS	Linux 2.6.9-1.667(Fedora Core release 3)
コンパイラ	バージョン 3.4.2
コンパイラオプション	-O3 -mfpmath=sse -march=pentium4

う．剛性マトリクス K が与えられた時刻 t_0 から微小時間 δt が経過して時刻 t_1 における剛性マトリクスが K' になったとする．あらかじめ K^{-1} がわかっているときに，K'^{-1} を求めるのにどれだけ時間が必要とされるのかについての評価を行う．式 (21) において，変化させる δK の形を変更したいくつかのモデルに対しての評価を行う．

剛性マトリクスには，医療情報部より提供していただいた大動脈のモデル (aorta) を用いる．大動脈の有限要素モデルとしては，4 種類のモデルを考える．つまり，

1. aorta197(ノード数 197，行列サイズ 438)
2. aorta388(ノード数 388，行列サイズ 900)
3. aorta799(ノード数 799，行列サイズ 1836)
4. aorta1661(ノード数 1661，行列サイズ 3765)

である．これらの有限要素モデルに対して Matrix Builder[4] を用いて生成された剛性マトリクスを K として用いる．ただし，Matrix Builder では直接法で変位計算を行うことができるように逆行列の形で生成される．そのため，逆変換してから SMW 公式を適用して逆行列を求めるための係数行列として利用する．また，剛性マトリクス K は正方行列であるのでサイズの二乗の要素数を持つ．

5.2.1 SMW 公式を用いる際に最低限必要とされる時間

SMW 公式を用いるうえで，どれくらい計算時間が必要とされるのかを知るために剛性マトリクスにおける最も小さな変化を考えてみる．速度上限を知るためにこのような実験を行う．Matrix Builder において，一つのノードが移動したと考えるのならば，一つのノードが情報として x, y, z の 3 つの成分を持っているので，剛性マトリクス K では 3×3 の大きさだけ変化したと考えることができる．そこで，表 3 に 4 種類のモデルそれぞれに対して，高速化手法を組

み合わせて得た結果を載せる．組み合せは，

1. SMW 公式のみで高速化手法は用いない
2. SMW 公式に加えて転置行列を作成
3. SMW 公式に加えて CRS 形式による疎行列性を利用
4. SMW 公式に加えて転置行列の作成と CRS 形式による疎行列性の利用
5. SMW 公式に加えて転置行列の作成と CRS 形式と CCS 形式による疎行列性の利用

のようになっている．表 3 と比較するために，野田らが行った aorta に共役勾配

表 3: 一つのノードが移動した場合

	aorta197	aorta388	aorta799	aorta1661
1	0.0109[s]	0.0485[s]	0.2058[s]	1.1213[s]
2	0.0073[s]	0.0311[s]	0.1801[s]	0.8283[s]
3	0.0061[s]	0.0265[s]	0.1109[s]	0.4615[s]
4	0.0060[s]	0.0261[s]	0.2164[s]	0.4477[s]
5	0.0050[s]	0.0218[s]	0.0894[s]	0.3777[s]

法を用いた場合において，解を求める際に必要とした時間計測の結果を表 4 に載せる [10]．aorta388 については，用いた力のデータがなかったため割愛した．表のアルゴリズムは以下のようになっている．

SparseCG　通常の CG 法に対して係数行列のデータ構造を疎行列対応のリスト構造にしたプログラム．

Sparse ICCG　ICCG 法における係数行列を疎行列対応のリスト構造としたプログラム．

SparseLU　スカイライン法を用いた LU 分解を行うプログラム．スカイライン法とは，疎行列向け LU 分解高速化手法で非ゼロ要素の分解計算をスキップすることにより高速化する手法である．

5.2.2　行列の要素数を変化させる割合を変更させた場合

変化した要素数が大きくなると，計算に必要とされる時間は増大する．SMW 公式を用いることができる限界を調査するという目的で実験を行う．表 5 に全要素のうち x %が変化したものとしての時間計測を行った結果を示す．ただし，

表 4: 共役勾配法による解法時間

	aorta197	aorta799	aorta1661
SparseCG	0.055[s]	0.449[s]	1.769[s]
SparseICCG	0.023[s]	0.340[s]	0.900[s]
SparseLU	0.016[s]	0.516[s]	3.420[s]

この変化した成分はSMW 公式を用いるのに適した形であるもとの仮定する．ここでいう SMW 公式を用いるのに適した形とは，δK_c の大きさが，縦方向に長い行列となるように変化した場合である．行列の成分の変化のさせ方を，1 つのノードが動いた場合と同じように 3×3 の大きさの部分を 1 セットとして変化させていくのだが，その際に δK を同じ列から変化させていくものとする．つまり，δK を全ての要素の値が 0 でない列と全ての要素が 0 の列というようにわけることができるような形で変化させていく．

また，アルゴリズムには表 3 において最も高速であった「SMW 公式に加えて転置行列の作成と CRS 形式と CCS 形式による疎行列性の利用」を行ったプログラムを用いる．

表 5: 行列の全要素のうち x %が変化した場合

x	aorta197	aorta388	aorta799	aorta1661
0.1	0.005420[s]	0.023179[s]	0.104429[s]	0.586251[s]
0.5	0.006310[s]	0.035439[s]	0.224785[s]	1.667832[s]
1	0.008543[s]	0.054121[s]	0.384469[s]	3.191520[s]
3	0.016125[s]	0.127614[s]	1.040893[s]	9.135045[s]
5	0.025308[s]	0.197247[s]	1.854894[s]	15.713603[s]
10	0.045612[s]	0.442656[s]	3.945401[s]	34.822260[s]

次に，δK_c の形による SMW 公式の効果について比較する．行列の全要素のうち，10 %が変化した場合における δK_c の形を以下のように設定して実験を行う．正方形に近い形とは，δK_c のサイズが $\frac{1}{\sqrt{10}}n \times \frac{1}{\sqrt{10}}n$ の正方行列となるように変化させた場合の形である．SMW 公式に最も適さない形とは δK_c のサイズが $\frac{1}{10}n \times n$ となるように変化させた場合の形である．

1. SMW 公式を用いるのに最適な形
2. 正方形に近い形
3. SMW 公式を用いるのに最も適さない形

表 6 に結果を示す.

表 6: SMW 公式を用いる行列の形による時間比較

	aorta197	aorta388	aorta799	aorta1661
パターン 1	0.045612[s]	0.442656[s]	3.945401[s]	34.822260[s]
パターン 2	0.152508[s]	1.555103[s]	14.541614[s]	139.249043[s]
パターン 3	1.967762[s]	19.535551[s]	175.785654[s]	2316.101325[s]

次に表 7 に行列の全要素のうち x %が変化した際に，関数ごとにかかった時間を比較する．表 3 において最も高速であった「SMW 公式に加えて転置行列の作成と CRS 形式と CCS 形式による疎行列性の利用」を行ったプログラムを用いる．行列の次数が小さいと差がわかりづらいため，最も大きなノード数を持つ aorta1661 で調査を行う．1〜10 は，式 (34) において，以下のような計算順序で計算している．演算子が書いてある部分がそのステップで計算している場所である.

1. E_c について CRS 形式に用いるための構造体を作成
2. $E_c \times K^{-1}$(CRS 形式)
3. δK_c について CCS 形式に用いるための構造体を作成
4. $K^{-1} \times \delta K_c$(CCS 形式)
5. $E_c \times K^{-1}\delta K_c$(CRS 形式)
6. $I + E_c K^{-1}\delta K_c$
7. $(I + E_c K^{-1}\delta K_c)^{-1}$ を直接法で計算
8. $(I + E_c K^{-1}\delta K_c)^{-1} \times E_c K^{-1}$
9. $K^{-1}\delta K_c \times (I + E_c K^{-1}\delta K_c)^{-1} E_c K^{-1}$
10. $K^{-1} - K^{-1}\delta K_c (I + E_c K^{-1}\delta K_c)^{-1} E_c K^{-1}$

また，これらの計算を行う際の，乗算のステップにおけるサイズを図 3〜図 7 に示す.

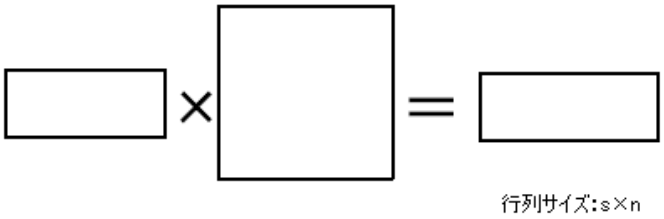

図 3: ステップ 2

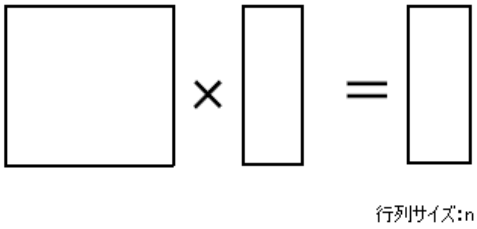

図 4: ステップ 4

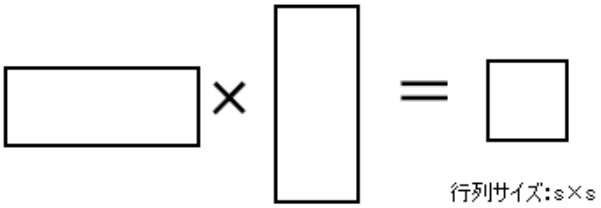

図 5: ステップ 5

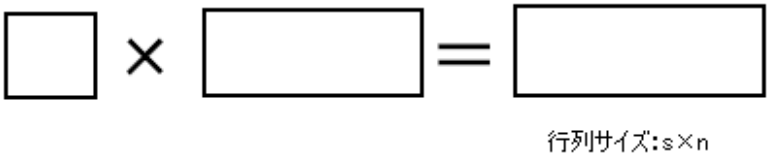

図 6: ステップ 8

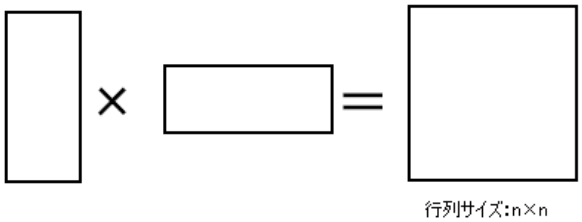

図 7: ステップ 9

表 7: 行列の全要素のうち x %が変化した場合の計算ステップ比較

ステップ	0.1	0.5	1	3	5	10
1	0.000087[s]	0.000285[s]	0.000514[s]	0.001523[s]	0.002297[s]	0.004547[s]
2	0.000315[s]	0.001103[s]	0.002012[s]	0.005789[s]	0.009723[s]	0.019395[s]
3	0.000262[s]	0.001220[s]	0.002615[s]	0.008732[s]	0.015467[s]	0.032539[s]
4	0.158383[s]	0.794833[s]	1.755486[s]	5.231959[s]	8.707520[s]	17.362042[s]
5	0.000084[s]	0.000099[s]	0.000115[s]	0.000265[s]	0.000556[s]	0.002039[s]
6	0.000005[s]	0.000009[s]	0.000016[s]	0.000125[s]	0.000418[s]	0.001571[s]
7	0.000007[s]	0.000108[s]	0.000550[s]	0.011967[s]	0.062007[s]	0.672943[s]
8	0.000500[s]	0.003976[s]	0.012290[s]	0.388892[s]	1.311393[s]	5.490475[s]
9	0.275752[s]	0.715648[s]	1.267941[s]	3.318674[s]	5.453751[s]	10.882542[s]
10	0.150856[s]	0.150551[s]	0.149981[s]	0.167119[s]	0.150471[s]	0.354167[s]
合計	0.586251[s]	1.667832[s]	3.191520[s]	9.135045[s]	15.713603[s]	34.822260[s]

5.2.3 MPI を用いた並列化による速度比較

SMW 公式を用いる際に，表 7 のステップ 9 において密行列の乗算を行う必要がある．この部分を並列化することで，計算時間の改善が行われたかどうか調査するために実験を行う．表 8 にてクラスタを用いて並列化した際にかかった時間を比較する．行列の全要素のうち，10 %が変化したものとする．評価する際のアルゴリズムには，表 3 において最も高速であった「SMW 公式に加えて転置行列の作成と CRS 形式と CCS 形式による疎行列性の利用」を行ったプログラムを用いる．

表 8: 並列化

台数	aorta197	aorta388	aorta799	aorta1661
1	0.045612[s]	0.442656[s]	3.945401[s]	34.822260[s]
2	0.064208[s]	0.482931[s]	3.793764[s]	31.041098[s]
4	0.070915[s]	0.486025[s]	3.603062[s]	29.228872[s]
8	0.078289[s]	0.495821[s]	3.518160[s]	28.025540[s]

次に，表 9 に，表 8 において，クラスタを 8 台用いた場合における計算ステッ

プごとに計測した時間を載せる．表7にて用いた計算ステップに加えて，ステップ11をクラスタ間のデータ通信とする

表9: 並列化における計算ステップ比較

ステップ	aorta197	aorta388	aorta799	aorta1661
1	0.000069[s]	0.000310[s]	0.001152[s]	0.004463[s]
2	0.000568[s]	0.002344[s]	0.009531[s]	0.040150[s]
3	0.000399[s]	0.001848[s]	0.008020[s]	0.037117[s]
4	0.026434[s]	0.238893[s]	2.046558[s]	17.582162[s]
5	0.000072[s]	0.000254[s]	0.001046[s]	0.003970[s]
6	0.000022[s]	0.000101[s]	0.000419[s]	0.001723[s]
7	0.001102[s]	0.006491[s]	0.057076[s]	0.664434[s]
8	0.010203[s]	0.075515[s]	0.563149[s]	5.550883[s]
9	0.002795[s]	0.022570[s]	0.164852[s]	1.388184[s]
10	0.000306[s]	0.001306[s]	0.005264[s]	0.022209[s]
11	0.036319[s]	0.146189[s]	0.661093[s]	2.730245[s]
合計	0.078289[s]	0.495821[s]	3.518160[s]	28.025540[s]

第6章　考察

6.1　高速化手法の効果確認

6.1.1　転置行列の作成の効果

表3から，転置行列の作成により配列の局所性を利用したプログラムでは，どのaortaにおいてもSMW公式のみを適用したプログラムよりも高速になっている．行列のサイズにかかわらず，転置行列を用いない場合と比較して3割程度高速なアルゴリズムとなっている．また，転置行列を作成したプログラムと，転置行列の作成に加えて疎行列性を利用したプログラムでは，わずかながら疎行列性も合わせて利用したプログラムのほうが高速化されている．つまり，疎行列性と配列の局所性は組み合わせることが可能であるが，その効果はそこまで大きいものではないといえる．

6.1.2　疎行列性を利用することによる効果

表 3 から，CRS 形式を利用したプログラムは，SMW 公式のみを利用したプログラムに対して，どの aorta においても高速化されている．これは，行列のサイズ n に対して変化させた部分のサイズ s が非常に小さいので，δK_c の成分がほとんど 0 となっていて，本来行わなければならない計算量と比較して大部分の計算を省くことが可能となっているためである．そして，どの aorta においても CRS 形式のみを用いたプログラムよりも，CRS 形式に加えて CCS 形式を用いたプログラムのほうが高速である．これは，行方向のデータ構造を圧縮することと列方向のデータ構造を圧縮することを並列に扱うことが可能であることを示しているとともに，疎行列性を利用することは，高速化に多大な影響を及ぼすということを示している．

6.2　一つのノードを動かした場合における反復法との速度比較

同じような場合における反復法の評価として，野田らが同じ aorta を用いて行った反復法を用いた解の導出結果と比較してみる．今回表 3 において行ったことは解の導出ではなく，逆行列の導出であるので単純に比較するということはできないが，この時点で反復法より速度が劣っていては SMW 公式を高速化に用いることができなくなってしまうので，SMW 公式の優位性を示すために示した．

表 3 と表 4 を比較すると，どの aorta においても SMW 公式を用いたプログラムのほうが大幅に高速である．解を導出するためには表 3 で逆行列を求めた時間に加えて，求めた逆行列にベクトルを乗じるということをしなければならないが，加えた力のうち多くは 0 となっているのでそれほど大きな時間は必要としない．表 3 の前提条件である剛性マトリクスの小さな変化では，SMW 公式は反復法よりも高速である．

6.3　変化させる割合による時間変化

6.3.1　x の値による速度低下の比較

表 3 の 5 から，x の値が増加するに従って，SMW 公式の速度は低下していることがわかる．速度が低下する割合を考えると，ノード数が大きい aorta のほうがノード数が小さい aorta に比べて速度低下の割合が大きい．つまり，次数の大きな行列ほど変化した成分の割合が大きくなるにつれて SMW 公式の効果

が小さくなることがわかる．

今回，全要素のうち x %が変化したということで比較を行ったが，要素数という観点から考えてみれば全要素数はサイズ n の二乗に比例する．よって，変化した割合は同じ x %であるが，サイズ $a1$ および $a2$ の行列において x %を変化させたとすれば，変化した要素数の比は $(a1)^2 : (a2)^2$ となる．よって，変化した割合が同じであっても，変化した要素数の比は等しくないのである．このことから考えると，サイズが大きい剛性マトリクスに対して SMW 公式は向かないと考えられる．

6.3.2　計算のクリティカルセクションの調査

表 7 から，aorta1661 変化する割合が大きくなったときに，どこがクリティカルセクションとなっているのか調査を行う．まず明らかなのは，疎行列計算であり，計算に必要とされる時間は少ないと考えられるステップ 4 に多大な時間がかかってしまっているということである．これは，δK_c が疎行列になっていないことが原因であると思われる．

式 (23) において δK_c を作成する際に，全ての要素が 0 の列を取り除いているが，SMW 公式に適した形にするためにノードが変化した部分を同じ列に集中させているので，ステップ 4 の部分に密行列が表れてしまった．そのため，7 で直接法で逆行列を求める時間を減らすことはできるが，本来密行列が表れない場所に密行列が出現してしまったと考えることができる．同じく密行列の乗算を行わなければならないステップ 9 においても同様に多大な時間がかかっている．

密行列演算でも，ステップ 10 の減算にかかる時間は x の値によってほとんど変化しないので，変化する割合によっては問題にならない．aorta1661 において，x の値にかかわらず，ステップ 4 とステップ 9 において演算に必要とされる計算時間が大きな割合を占めていることがわかる．計算におけるクリティカルセクションはステップ 4 とステップ 9 であり，その原因は密行列の乗算である．

6.4　並列化に対する考察

表 8 からノード数が小さい aorta197，aorta388 においては並列化したプログラムのほうが遅くなってしまっているが，ノード数が大きい aorta799，aorta1661 においては並列化したプログラムのほうが高速である．

表 7 のステップ 8，9 と，表 9 のステップ 8，9 を比較することで並列化部分を比較する．行列の次数が小さい aorta において並列化したプログラムが遅くなっ

ているのは，並列化による高速化の効果よりもデータ通信による遅延の効果が大きいためであることがわかる．それと反対に行列の次数が大きい aorta においては，並列化による高速化の効果がデータ通信による遅延の効果よりも大きく，高速化されていることがわかる．これらのことから，ノード数，つまり行列の次数が大きくなってくると，並列化による高速化が有効であるいうことがわかる．

6.5　式の精度に対する考察

SMW 公式を適用するということは，直接法のみで K^{-1} を求める場合と比較して，複雑な式 (34) を適用して解を求めるということである．数学的には式 (34) は等式となっているが，計算機上での計算の特性から丸め誤差が出てきてしまい，等しくなくなっている．誤差が出てくるといってもわずかなものであるので，シミュレーションの間だけ計算を行うことを考えると問題のない誤差である．

第7章　まとめ

本稿では，切断手技等の切断を行う際に変形を伴う挙動をインタラクティブにシミュレートするために，有限要素モデルで実際に用いられる剛性マトリクスの逆行列の導出が短時間で複数回必要とされる，という前提のもと SMW 公式を用いて行った．さらに，SMW 公式を用いる際に転置行列の作成，疎行列性を利用するためのデータ構造の変更といった高速化を施して，それぞれの手法に対する評価を行った．実装したプログラムでは，どの aorta を用いても「SMW 公式に加えて転置行列の作成と CRS 形式と CCS 形式による疎行列性の利用」を行ったプログラムが最も高速であった．

また，同じアルゴリズムで本研究室のクラスタを用いた並列化も行ったが，その際に’aorta197’および’aorta388’ではクラスタ 1 台で，’aorta799’および’aorta1661’ではクラスタ 8 台で行った場合が最も高速であった．ノード数の大きいモデルにおいて SMW 公式を用いる際には，クラスタを用いた並列化は有効である．

しかし，リフレッシュレートを満たすことが可能なレベルの速度で逆行列の導出を行うことができるのは aorta197 において全要素のうち 0.1 %〜5 %，aorta388 において全要素のうち 0.1 %のノードが移動した場合のみであった．

今後，リフレッシュレートを満たしたシミュレータ計算の実現するために，??節で述べたような高速化手法を実装して，本稿で述べた高速化手法を用いた計算をリフレッシュレートへと近づけることが重要である．

謝辞

本研究を進めるにあたって，御多忙の中，丁寧な御指導と御助言をいただいた，富田 眞治教授に甚大なる謝意を表します．

本研究を進めるにあたって，真心のこもった御指導と御助言をいただいた森眞一郎福井大学教授に深く感謝いたします．

本研究を進めるにあたって，異なる研究グループにもかかわらず，熱心な御指導をいただいた嶋田 創助手，三輪 忍助手，中島 康彦奈良先端技術大学院大学教授に心よりお礼申し上げます．

本研究を進めるに当たって，日頃から有益かつ適切な助言をいただいた Revolver グループ本学修士過程 1 回生である，野田 裕介氏，川原 崇宏氏には特に感謝します．

本研究を進めるにあたって，同学であり，互いに切磋琢磨しあった学部 4 回生である，鈴木 一範，中山 英卓，山口 明徳に深く感謝致します．

本研究を進めるにあたって，こちらの様々な要望を聞いていただき，お忙しい中熱心な御指導までしていただいた奈良先端技術大学院大学の中尾 恵助手，京都大学医学部附属病院医療情報部の粂 直人日本学術振興会特別研究員 PD にも感謝の意を述べさせていただきます．

参考文献

[1] Nakao, M.: Cardiac Surgery Simulation with Active Interaction and Adaptive Physics-Based Modeling, 京都大学大学院情報学研究科博士論文 (2003).

[2] Kuroda, Y.: A Study on Virtual Reality based Palpation Simulator, 京都大学大学院情報学研究科博士論文 (2005).

[3] Kume, N.: Distributed Massive Simulation for Haptic Virtual Reality Based Surgical Skill Transfer, 京都大学大学院情報学研究科博士論文 (2006).

[4] 中尾恵, 黒田嘉宏: 実時間力学計算手法のライブラリ化と手術シミュレータ

の開発, 平成 14 年度 IPA 未踏ソフトウェア創造事業「デジタル・ヒューマンを実現する人間機能のモデリングとその応用ソフトウェア」研究報告会 (2003).

[5] 広田光一, 金子豊久: 柔らかい仮想物体の力覚表現, 情報処理学会論文誌 (1998).

[6] 三好俊郎: 有限要素法入門, 培風館 (1978).

[7] 渡辺隆史, 大谷淳, 棚沢順, 徳永幸生: 2 段階境界要素法を用いる三次元弾性物体の変形と移動の実時間アニメーション法, 電子情報通信学会論文誌 (2005).

[8] 二宮市三, 吉田年雄, 長谷川武光, 秦野寧世, 杉浦洋, 櫻井鉄也, 細田陽介: 数値計算のわざ, 共立出版, 東京 (2006).

[9] P. パチェコ: MPI 並列プログラミング, 培風館 (2001).

[10] 野田裕介: 共役勾配法による手術シミュレータ高速化の予備評価, 情報処理学会関西支部大会 (2006).