2007－HPC－111 (35)
2007／8／3

数値計算アルゴリズム性能情報DBの自動チューニング技術への適用の検討

伊藤祥司
理化学研究所 情報基盤センター

線形方程式を例に取り上げて，数値計算アルゴリズムの体系的な性能比較の方法を提案する．この中で紹介する情報システムでは，典型的なテスト問題を用いて反復解法と前処理に対する求解性能情報のDBを生成し，線形方程式と求解アルゴリズムの間の性能比較を可視的に行う．本稿の中では，これらの方法を用いることによる効果を示す．さらに，自動チューニング技術に対して性能情報DBを適用するアイデアについても述べる．

A Study to Apply Performance Information Database of Numerical Algorithms to Automatic Tuning Technologies

Shoji ITOH
Advanced Center for Computing and Communication, RIKEN

Systematical performance evaluation methods for comparing numerical algorithms for linear equations are proposed. An introduced computational system generates a performance information database for a range of iterative solvers and preconditioning using a lot of typical test problems, and presents data visually allowing the relationships between the various algorithms and problems to be compared. In this paper, some effects by using these methods are shown. Further, some ideas to apply performance database to automatic tuning are also described.

1. はじめに

自然現象や工学現象の解明では，数値シミュレーションを用いた解析が盛んである．それらのシミュレーションでは，多くの場合，大規模で疎な$n \times n$ の係数行列を持つ線形方程式

$$Ax = b \tag{1}$$

を解くことに帰着される，シミュレーションに要する時間の多くがこの計算に費やされる．

ところが，線形方程式の求解アルゴリズムには様々なものが存在し，対象とする問題の性質によっては，その性能が十分に発揮されないような場合や数値解が得られない場合もある[1]．従って，実際のシミュレーションにあたっては，どの求解アルゴリズムを適用したら良いか指針が欲しい．

このような状況を改善するためには，実際の数値計算から得られるアルゴリズムの性能を示すデータ自体に注目し，それらを蓄積して分析することが重要である．これまでに，このようなデータに基づいて求解アルゴリズムの性能や特性を評価する方法を検討し，体系的に比較評価する取組みが行われてきている[2],[3].

本稿では，これらのアプローチをとおして，求解性能を表すデータを体系的に比較する方法２種類を提案する．その内の１つの方法により，反復解法の求解効率の影響は，解法よりも前処理の方が大きいことが確認された．また，本手法により見出される，I+S前処理に関する興味深い実験結果にも注目する．さらに，本研究の中で構築される数値計算アルゴリズム性能情報DBの自動チューニング技術への応用についても説明する．

2. 求解アルゴリズムに対する性能評価

任意の線形方程式(1)に対する求解アルゴリズムの分類としては，ガウス消去法に基づいたLU分解をはじめとする直接解法，SOR法などの定常反復解法，共役勾配(CG)法やBiCG法に代表される非定常反復解法が挙げられる．さらに，これらの求解効率を向上させるために併用する前処理などの技法も多く存在する．

このように，一つの方程式を解くための選択肢が多

いにも関わらず，解くべき線形方程式と求解アルゴリズム（本研究では，解法，および，解法と併用する前処理などの技法の組合せを指して求解アルゴリズムと呼ぶ）との間の相性などを示す情報が少ない．したがって，実際のデータに基づいて体系的に整理された情報による比較評価が重要である．

本研究では，求解する問題とアルゴリズムとの間の相関などを見出すことを念頭に置いて，求解すべき問題の特性を表すデータや，実際の計算を行ったときに求解アルゴリズムが生成するデータ，計算結果のデータなどに注目し，統計分析，データマイニングや品質管理などのデータ分析手法による評価分析や知識発見を行う．すなわち，計算環境を同一条件にした下で得られる，実際のデータに対する比較評価や特性分析などをとおして体系的な評価である．そのために，数値計算の様々な分野で取り扱う求解すべき問題と，求解によって得られる様々なデータとの定性的・定量的な分析を行うための，数値計算アルゴリズムに対して体系的にサーベイ・評価するシステムを開発している．ここでの評価システムとは，データ分析による評価手法，および，評価を行う情報システムとしての処理系の双方を指している．現段階では，線形方程式に対する反復解法に基づく求解アルゴリズムについての評価システムの開発を始めており，求解の際に得られる様々なデータに対する比較評価を行っている．

線形方程式に関する性能分析に関連する研究としては，線形解法などのパラメタを推定する，J. DongarraらによるSALSAプロジェクト[4]や，線形方程式の係数行列を分析して演算量を最小化するオーダリング方法や直接解法を提供するGrid-TLSEなどがある[5]．一方，本研究では，データ分析を用いて線形方程式や数値計算アルゴリズム自体の性能評価や新しい知見を見出すことが特徴である．

3. 性能情報DBの構築環境について

線形方程式を高速かつ高精度で解くために，様々な解法や前処理がこれまで提案されている．それらを集めて提供しているライブラリは多数あるが，本研究では，Lis（Library of Iterative Solvers for Linear Systems）[6] の逐次版を使用した．バージョンはlis-AMG-1.0.2修正版である．

Lisで用意されているアルゴリズム中で用いられるパラメタ値は，SAINVのドロップ値のみデフォルト以外の0.5を用い，それ以外はLisのデフォルト値を用いた．

線形方程式求解用アルゴリズムの開発などでは，典型的なテスト問題として様々なものがあるが，本研究では，これまでに，Matrix Market[7]の中から線形方程式向きのテスト行列を用いて問題を用意し，Lisを用いて解いた．右辺項のベクトルは，解ベクトルの全要素を1.0とし(1)式に代入して生成した．各アルゴリズムに与える初期ベクトルには，零ベクトルを用いた．収束判定は，アルゴリズム中の残差ベクトルに対する相対残差ノルムが$\|\boldsymbol{r}_k\|_2/\|\boldsymbol{b}\|_2 \leqq 1.0\times10^{-12}$を満たしたときとした（$\boldsymbol{r}_k$ はアルゴリズム中の残差ベクトル，k は反復回数）．最大反復回数には行列のサイズを用いた．これらの条件は，新しい数値計算アルゴリズムを提案した際の，数値実験でも採用される典型的なテスト問題の内の一つである．

これらを用いてジョブを実行し，データ採取するために構築した情報システムとして，計算サーバにSun Fire v880（CPU: UltraSPARC III, 900MHz，メモリサイズ: 16GB，OS: Solaris9，Sun Workshop 6 cc, f90）を用いた[2]．ここでは，アルゴリズムの基本的な情報を取得するために，ジョブ実行は全て1CPUで行った．コンパイルオプションは，LisのMakefileに記載されているデフォルトを用いた．

以上から，全線形方程式に対してLisの全アルゴリズムを用いて求解したとき，全て同一条件の計算環境における計算実験を実施できる．特に重要なことは，計算サーバの環境，すなわち，プロセッサ，メモリサイズ，コンパイラなどを常に同じ条件にした上で，ジョブ実行してデータ採取することである．

ここで得られた情報の中から，アルゴリズムの性能評価に有用なものをDBに格納した．具体的には，反復法のアルゴリズムの中で用いられている残差ベクトルのノルムの履歴情報と，数値解($\hat{x}$)が得られるまでに要した反復回数，CPU時間および数値解を用いて評価した真の残差

$$\hat{r} = b - A\hat{x} \tag{2}$$

のノルムの情報が，全行列と全求解アルゴリズムの全ての組み合わせについて格納されている．

4. 求解アルゴリズムの体系的な性能比較

本節では，DBに格納されている全てのデータに対して，相対的な性能比較を行い，その結果を視覚的に表示する体系的な性能比較の方法を提案し，それを用いて新たに確認できる知見について説明する．

4.1 求解性能の良いアルゴリズムに対する比較

１つの係数行列から構成される線形方程式について，Lisの全アルゴリズムを用いて得られた求解結果に対し，収束までの所要反復回数が最小であったアルゴリズムと，それ以外のアルゴリズムとの比を算出し，その数値を指標として性能を比較した．つまり，各々の求解アルゴリズムに対して，

$$\text{Score} = 10 \times (\text{最小の所要反復回数} \div \text{他のアルゴリズムの所要反復回数}) \quad (3)$$

を算出した．右辺冒頭の“10×”は，Scoreを２桁の整数に揃えるための係数であり，指標の意味は変わらない．また，式(3)は所要反復回数の値をCPU時間に置換えても同様な評価を行える．

式(3)による相対的な性能指標を用いて，DB内の求解性能データの性能比較を行った．ここでは，Matrix Marketの中から線形方程式向きの下記52個の行列を用いた．

{1138, 494, 662, 685}_bus, add{20, 30}, arc130, bcsstk{14, 15, 16, 17}, bcsstm26, fs_183_{1, 3, 4, 6}, fs_541_[1-4], fs_680_[1-3], fs_760_[1-3], gr_30_30, gre_1107, gre_216{a, b}, gre__{115, 185, 343, 512}, jpwh_991, lund_{a, b}, memplus, nos[1-7], s1rmq4m1, s1rmt3m1, s2rmq4m1, s2rmt3m1, s3rmq4m1, s3rmt3m1, s3rmt3m3

各々の行列から得られる線形方程式を，Lisの全アルゴリズムを用いて解いたときの所要反復回数に対して，式(3)でScoreを算出した．この値に応じて，Fig.1のように色分けして図示したものが，Fig.2, 3である．

Figure 1　Score と Color の対応付け

4.2 本研究手法による知識発見

Fig.2 では，12種類の解法ごとにグループ分け（太枠）しており，Fig.3 では，8 種類の前処理ごとにグループ分けしている．従って，一つの線形方程式に対して収束までの所要反復回数が少なかったアルゴリズムは，黒，赤などの濃い色で表現され，所要反復回数が多いほど，黄，薄黄，白と薄い色で表現されるため，全アルゴリズムの求解性能の傾向を体系的，視覚的に確認できる（実際のシステムでは色による識別が可能だが，本稿ではモノクロのため，グレースケールの濃淡による表現となっている）．

また，「全く収束していない」場合にはドットを，式(2)による真の残差ベクトルのノルムが10^{-8}以下であった「見かけ上の収束」の場合にはアスタリスクを印している．

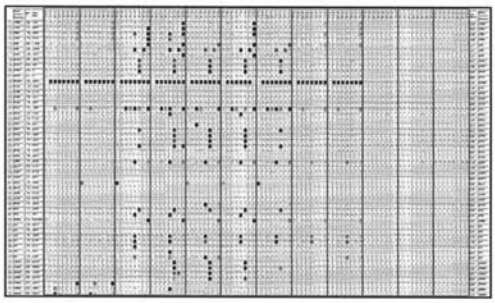

Figure 2　解法ごとにグループ分けしたSurvey chart

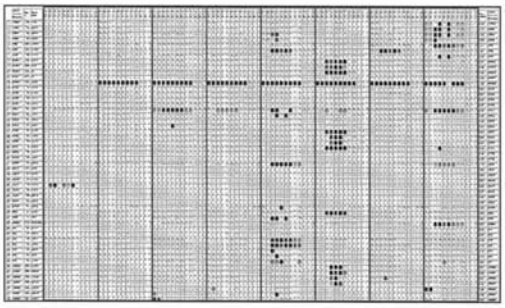

Figure 3　前処理ごとにグループ分けしたSurvey chart

Fig.2, 3を比較すると，反復解法に基づく求解アルゴリズムの効果は，解法の違いよりも前処理の違いによる影響の方が顕著であることが確認できる．

ここで，I+S前処理の効果に注目した．Fig.3のデータを係数行列が非対称なもの（26種類）と対称なもの（26種類）とに分けてまとめた結果が，Fig. 4〜7 である．

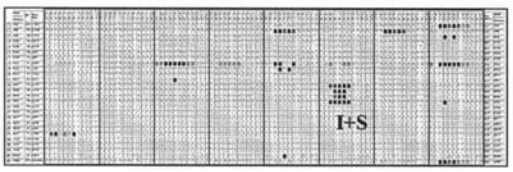

Figure 4 非対称系に対する結果(所要反復回数)

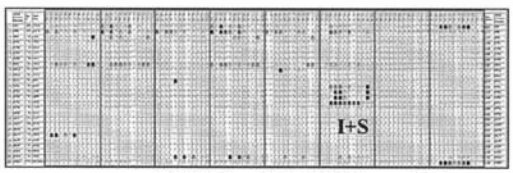

Figure 5 非対称系に対する結果(CPU時間)

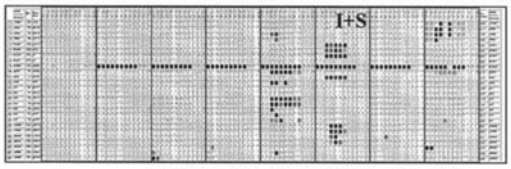

Figure 6　対称系に対する結果(所要反復回数)

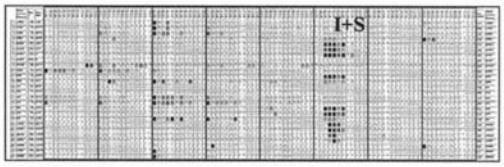

Figure 7　対称系に対する結果(CPU時間)

I+S前処理は，元々はGauss-Seidel法などの定常反復法に対して開発された技法であるが[8]，河野らによって，その改良[9]や非定常反復法に対する適用についても研究されてきている．

I+S前処理では，元々の線形方程式(1)に対して以下のような変換が行われる．

係数行列

$$A = \left[a_{i,j} \right], \quad (i, j = 1, 2, \cdots, n) \tag{4}$$

に対して，その対角要素からなる対角行列D を生成し，

$$D^{-1}Ax = D^{-1}b. \tag{5}$$

とあらかじめ変換する．

そして，行列A の対角要素の右側に位置する要素をm個取り出して各々の符号を反転させた行列S

$$S^{(m)} = \begin{bmatrix} 0 & -a_{1,2} & \cdots & -a_{1,m+1} & 0 & \cdots & \cdots & 0 \\ \vdots & 0 & -a_{2,3} & \ddots & -a_{2,m+2} & & & \vdots \\ \vdots & & 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & & & \ddots & \ddots & \ddots & -a_{n-m,n} \\ \vdots & & & & & \ddots & \ddots & \vdots \\ \vdots & & & & & & \ddots & -a_{n-1,n} \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \end{bmatrix} \tag{6}$$

と，単位行列I を用いて，各反復計算では

$$(I + S^{(m)})\tilde{A}x = (I + S^{(m)})\tilde{b} \tag{7}$$

と前処理された線形方程式を解く．ここで，$\tilde{A} = D^{-1}A$，$\tilde{b} = D^{-1}b$ である．本実験の中では，Lisライブラリのデフォルト値にしたがってm=3とした．このように，I+Sは，前処理行列の生成など関連する演算に要する計算コストの低い前処理である．

以上から，係数行列A が対称行列であった場合でも，I+S前処理を施した系は非対称となる．一般に，対称行列を係数行列にもつ線形方程式に対しては，CG法など，その対称性を維持するような求解アルゴリズムを用いる．ところが，本実験条件下においては，I+S前処理により対称性が崩れた系に対しても，求解に要する反復回数やCPU時間において，高い性能を示す事例が確認できた．これは，理論面からのアプローチでは見出しにくく，本手法を用いることににによって始めて確認できる知見の一つである．

4.3 求解性能の悪いアルゴリズムに対する比較

一方で，性能の悪い方からのアルゴリズムの比較評価も行った．ここで新たに用いる評価指標(Score2)の一つとして，

$$\text{Score2} = 10 \times \left(\text{他のアルゴリズムの所要反復回数} \div \text{最大の所要反復回数} \right)^2 \tag{8}$$

を用いた．

この評価式では，線形方程式を求解できたアルゴリズムについて，その所要反復回数を用いて計算している．最も遅く収束した（所要反復回数が一番多い）求解アルゴリズムに対しては，Score2=10であり，他のアルゴリズムのScore2は相対的に小さくなる．式(8)は，(3)同様に，CPU時間についても評価可能である．

式(3)(8)の評価式をグラフに表したものが，Fig. 8である．縦軸がScoreおよびScore2の値を，横軸が所要反復回数の割合を示している．実際には，少数以下を四捨五入し整数としているので，実際のScore，Score2のグラフは階段状であるが，ここでは評価の概要を表すために簡略化して表示している．

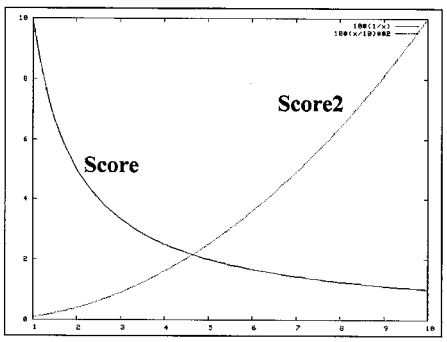

Figure 8 各評価式のグラフ

Fig. 8では，横軸の小さい値の領域でScoreの傾きが大きい．逆に，縦軸の大きい値の領域ではScore2の傾きが大きくなる．つまり，Score は性能の良いアルゴリズムの情報を目立たせ，Score2 は性能の悪いアルゴリズムの情報を目立たせている．

5. 自動チューニング技術への適用

近年の研究では，数値計算プログラムも含めたソフトウェア工学の領域にも及ぶ，自動チューニングが注目されている．その主な研究目的は，異なるハードウェアに対して，ソフトウェアの移植を簡単にするための取り組みである．

5.1 片桐らによる自動チューニング研究への取組み

その中心の研究は，片桐らによるソフトウェア自動チューニングであり，その取組みの中ではソフトウェアとしての数値計算ライブラリの性能を，各種計算機アーキテクチャに応じて最大限に引き出すよう自動的に最適化することを主眼とした研究が行われている[10]．今村らによる研究では，ソフトウェアの性能測定にも注目した数値計算ソフトウェア開発の研究が展開されている[11]．小谷，須田らは，品質管理の中の一手法としても位置付けられる実験計画法を，自動チューニングにおける性能評価関数の推定へ応用する研究に取組んでいる[12]．

5.2 直野らによる数値計算ポリシーの提案

直野らによる数値計算ポリシー入力型の行列計算ライブラリ方式では，ライブラリユーザの要求用件に応じた求解アルゴリズムの指針を提示する方式が研究されている．その一提案として，ベクトル群の直交化アルゴリズムを例として，ユーザの要求が計算精度と計算速度のどちらにあるかをポリシー精度 ε により判定し，計算するライブラリがある[13]．さらに，計算機アーキテクチャに応じて最適なアルゴリズムが異なることにも着目し，異種混合の計算機環境での性能評価を行い，数値計算ポリシーによって，より良いアルゴリズムが選択されていることを確認している[14]．

5.3 本研究における取組み

本研究で構築されたDBの性能情報は，これらの自動チューニングの研究に対して，チューニングによる改善度を評価するための基本データとなる，あるいは，要件に応じたアルゴリズムを選択するための指針となる情報を提供することも可能である[15]．また，直野らが提案した数値計算ポリシー方式を線形方程式にも適用する際には，求解性能DBに蓄積された(2)式から得られるデータを用いて，ポリシー ε によるアルゴリズム切替えの指針となる情報を提供できる．特に，4.3節で説明した評価情報は，選択すべきではないアルゴリズムの見通しを予め提示するものである．

さらに，様々なハードウェア環境下で測定された性能情報自体をDBに蓄積していくことで，同じ求解アルゴリズムやテスト問題の組み合わせであっても，マシンアーキテクチャの違いによる性能のズレを分析評価することに利用できる．こちらについては，片桐，今村，須田らの方法に基づくデータをDBに蓄積することも考えられる．

6. まとめ

本稿では，線形方程式に対する求解アルゴリズムの例を挙げて，その利用における現状および問題点について述べた．それら問題点の解決に向けてのアプローチの１つとして，多くのテスト問題を解いて得られる求解性能のデータに着目し，求解アルゴリズムを体系的に性能評価する方法について説明した．その中で，求解性能の良い方からのアルゴリズムに対する性能比較方法と，求解性能の悪い方からの比較方法の２種類を提案した．

性能情報DBの構築に当たっては，Matrix Marketで提供されている線形方程式向きの行列を用いてテスト問題を生成し，Lisライブラリに用意されている全アルゴリズムを適用して，３節で説明した方法で実施した．そのデータ分析から，本実験の条件下において，反復解法に基づく求解アルゴリズムの効果は，解法よりも前処理の違いにおける影響の方が顕著に表れていることが確認された．また，I+S前処理は係数行列の対称性を崩すにも関わらず，対称系の線形方程式に対する効果も確認された．

これらは，理論面からの考察のみでは見出すことが難しい事柄であり，データマイニングにおける“知識発見”の一つとして，本研究でのアプローチだからこそ得られる知見である．これは，本論文で提案している，同一条件の計算環境下での実験データに基づく体系的な評価によってはじめて見出せることである．

更に，この様な実行性能に基づくアルゴリズムの性能情報DBの自動チューニングへの利用についても説明した．

どの問題に対して，どのアルゴリズムが有効であるかという議論は，今後の課題である．

参考文献

[1] Barrett, R., et. al., *Templates for the solution of linear systems: Building Blocks for Iterative Methods*, SIAM, 1994. （邦訳）長谷川里美，長谷川秀彦，藤野清次：反復法 Templates，朝倉書店，1996.

[2] 伊藤祥司，線形方程式求解アルゴリズムに対する体系的な性能比較について，情報処理学会研究報告，2006-HPC-107, pp.281-286, (2006).

[3] Itoh, S., Comparison of Effects of Solver and Preconditioning for Linear Equations Based on Systematical Data Analysis, Proc. of Int. Conf. on Comput. Method (ICCM2007), Hiroshima, Japan, Apr., (2007).

[4] SALSA Project, http://icl.cs.utk.edu/salsa/

[5] Grid-TLSE project, http://www.enseeiht.fr/lima/tlse/

[6] Kotakemori, H., Hasegawa, H. and Nishida, A., Performance Evaluation of a Parallel Iterative Method Library using OpenMP, In proceedings of the 8th International Conference on High Performance Computing in Asia Pacific Region (HPC Asia 2005), pp.432-436, 2005. http://ssi.is.s.u-tokyo.ac.jp/lis/

[7] Matrix Market, http://math.nist.gov/MatrixMarket/

[8] Gunawardena, A.D., Jain S.K. and Snyder L., Modified iterative methods for consistent linear systems, Linear Algebra Appl., 154-156, pp. 123-143, (1991).

[9] Kohno T., Kotakemori H. and Niki H., Improving the Modified Gauss-Seidel Method for Z-matrices, Linear Algebra and its Applications, 267, pp. 113-123, (1997).

[10] 片桐孝洋，ソフトウェア自動チューニング，彗文社，2004.

[11] 今村俊幸，性能測定基盤と連携する数値計算ソフトウェア，情報処理学会研究報告，2006-HPC-107, pp.199-202, (2006).

[12] 小谷和正，須田礼仁，汎用的なソフトウェア自動チューニング機構のための実験計画法の応用の検討，情報処理学会研究報告，2006-HPC-107, pp.193-198, (2006).

[13] 直野 健，猪貝光祥，木立啓之，数値計算ポリシーを入力とするベクトル群の直交化ライブラリ，情報処理学会論文誌：コンピューティングシステム，Vol.46, No.SIG 7(ACS10), pp.35-43 (2005).

[14] 直野 健，猪貝光祥，木立啓之，数値計算ポリシー入力型グラムシュミット直交化ライブラリの異種混合計算機環境における性能評価，情報処理学会論文誌：コンピューティングシステム，Vol.46, No.SIG 12(ACS11), pp.279-287 (2005).

[15] 伊藤祥司，数値計算アルゴリズム評価のための性能情報 DB 構築と自動チューニングにおける利用，計算工学講演会論文集，12(2)，pp.567-570，(2007).