2008－HPC－116 (9)
2008／8／5

高速性と信頼性を両立するAC-IDR(s)法の提案と評価

櫻井隆雄，直野健，恵木正史，（株）日立製作所中央研究所
猪貝光祥，木立啓之，小路将徳，（株）日立超 LSI システムズ

要旨

科学技術計算において大規模行列の連立一次方程式の求解は最も時間を要する計算の 1 つであり，その高速な解法は常に求められている。近年，IDR(s)法という新たな連立一次方程式解法が提案された。この解法は従来のものより高速だが，稀に出力される解が要求精度を満たさずに偽収束する問題があった。本稿ではこの偽収束の原因が演算量削減を目的とする近似演算による誤差だと解明し，誤差の発生を事前予測して近似演算の使用を判断する自動チューニング方式を実装した Auto Corrected-IDR(s)法を提案した。標準の行列を用いた数値実験の結果，出力された解が偽収束せずに要求精度を満たした割合が従来法は 61%であるのに対し提案法は 100%を達成できた。

Proposal and Evaluation of AC-IDR(s) Method That Combines High-Performance with High Accuracy

Takao Sakurai, Ken Naono, Masashi Egi, Central Research Laboratory, Hitachi, Ltd.
Mitsuyoshi Igai, Hiroyuki Kidachi, Masanori Shoji, Hitachi ULSI Systems Co., Ltd.

Abstract

IDR(s) methods performs faster than conventional iterative solvers. However, the methods occasionally outputs incorrect solutions. Our purpose is to analyze the cause of the incorrect solutions. We focus the approximation step in IDR(s) of reducing the computational complexity. We propose an auto-tuning type IDR(s) method, which we call "Auto-corrected IDR(s) method" (AC-IDR(s)). The method predicts the occurrence of the incorrectness in the approximation step. Numerical experiments show that the results of AC-IDR(s) method are completely correct in all cases.

1. はじめに

電磁場や流体の解析などに代表される科学技術計算において，大規模な係数行列に対する連立一次方程式の求解は，最も計算時間を必要とする主要な演算である。そのため，連立一次方程式を高速に演算する方式は常に必要とされている。

本稿では，連立一次方程式解法の1つである IDR(s)法 (Induced Dimension Reduction (s) method)[2]を題材とする。IDR(s)法は Sonneveld と van Gijzen により最近提案された新しい Krylov 部分空間を用いた反復解法であり，従来連立一次解法として使われてきた BiCG 法[3]や GMRES(m)法[4]と比べて短い時間で解が求められることが実証されている[8]。IDR(s)法は他の解法と同様に様々なパラメータを持ち，その中でも Krylov 部分空間の次元数 s は特に重要なパラメータである。s を大きくすると演算に必要なメモリ量は増加し，1反復当りの演算時間は増加するが，収束性が向上する。よって，s が小さいときに収束しない行列に対して s を大きくすると収束する場合がある。しかし，この s の値を大きくすると IDR(s)法が出力した近似解と連立一次方程式の真の解の誤差が，使用者の入力した要求精度と比べて非常に大きくなる偽収束という現象がまれに発生する問題がある。この問題は常に入力された要求精度よりも誤差の小さい近似解を出力しなければならないという解法の要件を満たす上で障害となる。よって，この問題の解決が IDR(s)法の大きな課題となっている[5]。

そこで，本稿では IDR(s)法において上記の問題の原因を解明し，IDR(s)法の短い時間で解を求められる高い性能を維持した上で，偽収束の発生を抑制し常に要求精度を満たす解を得られる方式を実現することを目的とする。

2. IDR(s)法の概要とその問題点

2.1 IDR 定理

本節では IDR(s)法の前提となる IDR 定理を述べる。

$A \in R^{n\times n}$，行列 $P := (p_1, p_2, \cdots, p_s)$（$p_i (i = 1,2,\cdots,s)$ は一次独立なベクトル群），完全 Krylov 空間 $K_n(A, r_0)$ の空間 g_0 が存在し，空間 $g_j (j = 1,2,\cdots)$ を以下で定義する。

$$g_j := (I - \omega_j A)(g_{j-1} \cap \mathrm{NULL}(P^T)) \quad \cdots(1)$$

ω_j は非零のスカラー値，$\mathrm{NULL}(P^T)$ は P^T の零空間である。ここで，g_j の次元数を d_j，$I - \omega_j A$ が非特異の場合，以下の IDR 定理が成り立つ。

$$0 \le d_j - d_{j+1} \le d_{j-1} - d_j \le s \quad \cdots(2)$$

2.2 IDR(s)法のアルゴリズム

IDR(s)法のアルゴリズムについて述べる。IDR(s)法は従来連立一次方程式の解法として代表的な GMRES(m)法や BiCG 法と同様の Krylov 部分空間法である。Krylov 部分空間法は残差ベクトル $r_{k+1} (= b - Ax_{k+1})$ が Krylov 部分空間 $K_k(A, r_k)$ に属するように近似解ベクトル x_{k+1} を生成し，要求精度を下回るまでこれを繰り返すのが特徴である。IDR(s)法もこれに従い r_{k+1} が(1)式で定義した Krylov 部分空間 g_{j+1} に属するように x_{k+1} を生成する。IDR(s)法の特徴は，IDR 定理により g_{j+1} の次元数が単調減少するため，それに属する r_{k+1} が零ベクトルへ収束することである。

r_{k+1} が g_{j+1} に属するのはベクトル $v \in g_j \cap \mathrm{NULL}(P^T)$ に対して以下の条件を満たすことである。

$$r_{k+1} = (I - \omega_{j+1}A)v \quad \cdots(3)$$

ベクトル v は空間 g_j に属する過去 s 回分の残差ベクトル $r_{k'}(k' = k-s+1,\cdots,k)$ の組合せで表せる。

$$v = r_k - \sum_{j=1}^{s}\gamma_j e_{k-j} \ (e_i = r_{i+1} - r_i) \quad \cdots(4)$$

このとき，行列 $E_k := (e_{k-1}, e_{k-2}, \cdots, e_{k-s})$ を定義し，$E_n c = \sum_{j=1}^{s}\gamma_j e_{k-j}$ を満たすベクトル c を用いると(4)式は次のように変形できる。

$$v = r_k - E_k c \quad \cdots(5)$$

また，v は空間 NULL(P^T) に属するため，次を満たす。

$$P^T v = 0 \quad \cdots(6)$$

(5)(6)式より c は次の連立一次方程式で求められる。

$$P^T E_k c = P^T r_k \quad \cdots(7)$$

(3)式のパラメータ ω_{j+1} は $\|r_{k+1}\|_2$ を最小化するため，以下の値とする。

$$\omega_{j+1} = (v, Av)/(Av, Av) \quad \cdots(8)$$

一方，ベクトル e_k は次のように表せる。

$$\begin{aligned} e_k &= r_{k+1} - r_k \\ &= v - \omega_{j+1}Av - r_k \\ &= r_k - E_k c - \omega_{j+1}Av - r_k \\ &= -E_k c - \omega_{j+1}Av \end{aligned} \quad \cdots(9)$$

また，ベクトル $q_k = x_{k+1} - x_k$ は e_k と次の関係を持つ。

$$\begin{aligned} e_k &= r_{k+1} - r_k = (b - Ax_{k+1}) - (b - Ax_k) \\ &= -A(x_{k+1} - x_k) = -Aq_k \end{aligned} \quad \cdots(10)$$

さらに行列 $Q_k := (q_{k-1}, q_{k-2}, \cdots, q_{k-s})$ を定義すると，q_k は以下のように表せる。

$$\begin{aligned} q_k &= -A^{-1}e_k \\ &= -(-A^{-1})E_k c - (-A^{-1})\omega_{j+1}Av \\ &= -Q_k c + \omega_{j+1}v \end{aligned} \quad \cdots(11)$$

以上から，IDR(s)法のアルゴリズムは図 1 のようになる。

1 行から 8 行までが最初のベクトル v を生成するための準備にあたる部分で，9 行から 23 行がメインループとなる。10 行は s 元の連立一次方程式を解きベクトル c を求める部分であり，上記の(7)式に相当する。11 行は c からベクトル v を生成する(5)式に当たる。13，14 行は(8)式の ω を生成する演算である。15 行および 18 行は共に(11)式の q_k を求める演算である。

16 行と 19 行も共に e_k を求める演算であるが，演算内容が違うのは，行列 A とベクトルの乗算の演算量が行列 A の非零要素数回と非常に大きいため，16 行の処理では既に 13 行で求めている $t_k = Av_k$ を利用して $-E_k c_k - \omega t_k$ の近似演算をする方が，$-Aq_k$ を演算するよりも演算量が少ないからである。一方で，19 行の処理は Av_k を求めていないため，$-E_k c_k - \omega Av_k$ の近似演算よりも $-Aq_k$ を直接演算した方が演算量は少ない。

22 行は収束判定であり，残差ベクトルの 2 ノルムが要求精度 ε を下回った場合，反復を終了する。

1: Let x_0 be an initial guess, and put $r_0 = b - Ax_0$
2: Do $k = 0, s-1$
3: $v_k = Ar_k$
4: $\omega = \dfrac{(v_k, r_k)}{(v_k, v_k)}$
5: $q_k = \omega r_k$, $e_k = -\omega v_k$
6: $r_{k+1} = r_k + e_k$, $x_{k+1} = x_k + q_k$
7: CONTINUE
8: $E_s = (e_{s-1}, \cdots, e_0)$, $Q_s = (q_{s-1}, \cdots, q_0)$
9: Do $k = s, iter$
10: Solve c_k from $P^T E_k c_k = P^T r_k$
$\left[P = (p_1, \cdots, p_s),\ p_i = r_i / \|r_i\| \right.$
$\left. (p_i, p_j) = \begin{cases} 1 & (i = j) \\ 0 & (i \neq j) \end{cases} \right]$
11: $v_k = r_k - E_k c_k$
12: If mod$(k, s+1) = s$ then
13: $t_k = Av_k$
14: $\omega = \dfrac{(t_k, v_k)}{(t_k, t_k)}$
15: $q_k = -Q_k c_k + \omega v_k$
16: $e_k = -E_k c_k - \omega t_k$
17: Else
18: $q_k = -Q_k c_k + \omega v_k$
19: $e_k = -Aq_k$
20: End If
21: $r_{k+1} = r_k + e_k$, $x_{k+1} = x_k + q_k$
22: If $\|r_{k+1}\|_2 / \|b\|_2 \leq \varepsilon$ then Exit Do
23: CONTINUE

図 1：IDR(s)法のアルゴリズム

2.3 パラメータ s の増大により発生する IDR(s)法の問題

IDR(s)法におけるパラメータ s は Krylov 部分空間 g_{j+1} の初期次元数とベクトル v の生成頻度を決定するパラメータである。s が大きくなると g_{j+1} の次元数が増加するため 1 反復あたりの解の探索範囲が広がり，収束に必要な反復回数が減少する。一方で，演算に使用する行列 E_k，Q_k の列数や正方行列 $P^T E_k$ の次元数が s であるため，s が大きくなると 1 反復当りの演算量が増加し，IDR(s)法全体の演算時間が増加する。また，演算に必要とするメモリ量も s に伴い増加するという問題がある。

さらに，s が大きくなると，より深刻な問題が発生する。IDR(s)法に限らずあらゆる反復解法は，ソルバが収束とみなしたときに出力された近似解ベクトル x_n に対して，真の解との相対残差の 2 ノルム $\|b - Ax_n\|_2 / \|b\|_2$（以後，これを「真の相対残差 r_t」と呼ぶ）が必ず要求精度 ε を下回っていなければならない。しかし，IDR(s)法は，s が増加すると真の相対残差 r_t が要求精度 ε よりも大きくなる偽収束という現象が発生する問題がある。また，このときソルバの出力する残差 r_n のノルムは ε を下回っており，ライブラリ使用者は真の相対残差 r_t を求める検算をしなかった場合，偽収束の発生に気づかずに精度が不十分な解を別の演算に用いる恐れがある。図 2 にある行列 ex26 を入力としたと

きの各 s におけるソルバの出力した残差 r_n と真の相対残差 r_t それぞれのノルムの関係を示した。なお，この行列 ex26 は MatrixMarket[6]に掲示されたものである。s の増加に伴い r_t が増大していることが読み取れる。

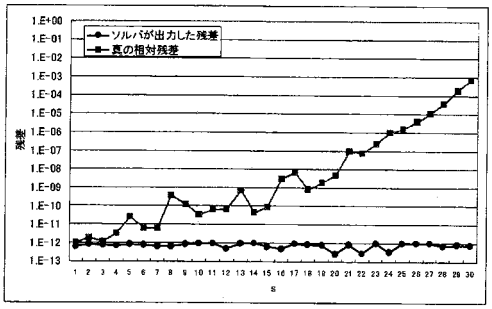

図 2:各 s における r_n と r_t（要求精度 $\varepsilon = 10^{-12}$）

IDR(s)法の原著論文[2]によると，内部パラメータ ω をある閾値を超えないように制限することで r_t の増加を抑える方法が記述されているが，この方法では完全に r_t の増加を防ぐことはできないという測定結果も発表されている[5]。

よって，本研究の目的はIDR(s)法における偽収束発生の原因を特定し，それを完全に抑制して高性能で出力される解が常に要求精度を満たす連立一次方程式解法を実現すること，とした。

3. 高速性と信頼性を両立する方式の提案

3.1 偽収束発生の原因とその対策による課題

2.2 節で述べたように IDR(s)法での k 回目の反復における残差 r_k の微小変化ベクトル e_k の算出法は，近似解ベクトル x_k の微小変化ベクトル q_k を用いて $-Aq_k$ の演算で求める直接演算と，演算量を削減するため式変換により同値となる $-E_k c_k - \omega t_k$ の演算により求める近似演算の 2 通りが存在する。

この 2 通りのうち，直接演算 $-Aq_k$ では e_k の算出が q_k に依存しているのに対し，近似演算では e_k の算出が $-E_k c_k - \omega t_k$ であり，q_k の算出が $-Q_k c_k + \omega v_k$ と相互に独立で行われている。この近似演算により e_k と $-Aq_k$ の間に誤差が生じる可能性があり，これが偽収束の発生に関連があると考えられる。

そこで，2 章の計測で用いた行列 ex26 を入力としたときの各 s について，収束までの $-E_k c_k - \omega t_k$ と $-Aq_k$ の近似誤差 $\|(-E_k c_k - \omega t_k) - (-Ap_k)\|_2 / \|b\|_2$ の演算開始から終了までの合計値を測定した。図 3 にその結果と真の相対残差 r_t の比較を示す。

図3から近似誤差の合計が r_t と一致していると読み取れる。よってIDR(s)法で偽収束が発生するのは微小変化ベクトル e_k 算出において演算量削減を目的として近似演算 $-E_k c_k - \omega t_k$ を用いて求めるためであり，常に直接演算 $-Aq_k$ を用いればこれを抑制できると考えられる。しかし，近似演算を用いていた理由は演算量を削減するためであり，それを用いない場合，ソルバ全体の演算量が増加し，求解性能が劣化するという新たな問題が生じる。また，IDR(s)法では近似演算はs+1反復に1回行われていたため，sが小さいときは r_t が十分小さいにもかかわらず，求解性能の劣化が顕著となると考えられる。そこで，偽収束発生につながるほど近似誤差が大きくなる頻度がどの程度か確認するため，改めて測定を行った。

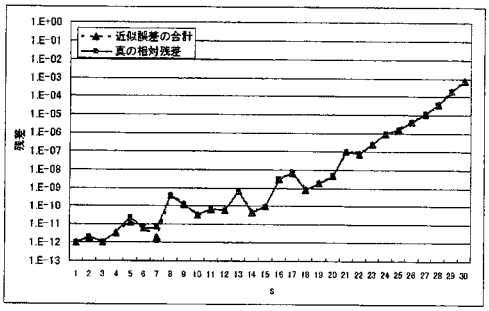

図3:近似誤差の合計と真の相対残差の関係

図 4 に IDR(s)法で行列 ex26 を入力とし，s=1～30 として解いたときの近似誤差の分布を示した。入力される要求精度 ε が通常は $10^{-8} \sim 10^{-12}$ 程度であること考慮すると，$-E_k c_k - \omega t_k$ と $-Aq_k$ の誤差が 10^{-12} 以上となり偽収束発生につながるケースは 7%程度と稀であり，大部分は e_k を近似演算で算出して問題がないと読み取れる。

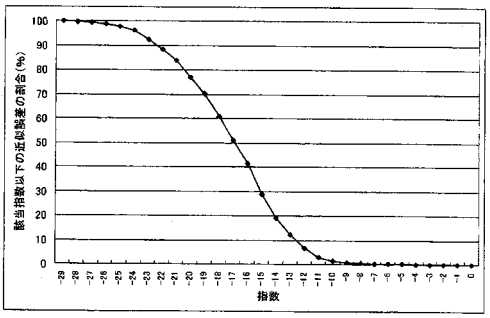

図 4:近似誤差の分布

そこで，何らかの偽収束発生を予測する何らかの指標を用いて適切に近似演算と直接演算を使い分ける実行時自動チューニング方式[1]を用いればIDR(s)法の高い演算性能を維持したまま常に要求精度を満たす解を出力できると考えられる。この実行時自動チューニング方式実現のためには，偽収束の発生を e_k 算出の前に予測する指標を策定する必要がある。この指標の策定が高性能と解の信頼性を両立させたIDR(s)法実現のための新たな課題となる。

3.2 偽収束発生を予測する指標の策定

本節では実行時自動チューニング方式に必要となる偽収束発生を予測する指標を策定する。IDR(s)法は s が大きくなると，Krylov 部分空間の次元数が増大し，探索空間

が広大になるため収束が早まるが，相対残差ノルム $\|r_k\|_2/\|b\|_2$ が反復回数の少ないときには一時的に非常に大きな値をとる場合がある。図5に行列ex26，s =10における相対残差ノルムの履歴を示す。図5の場合，相対残差ノルムが一時的に 10^4 を超えているとわかる。

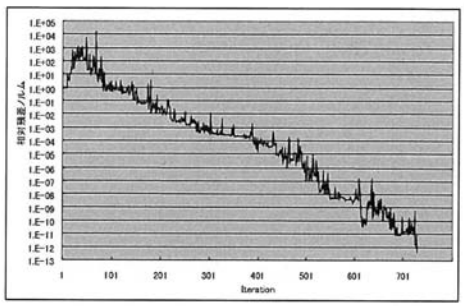

図5:行列 ex26，s =10 の相対残差ノルムの履歴

この高い相対残差ノルムが $-E_k c_k - \omega t_k$ と $-Aq_k$ の近似誤差に影響しているか確認するため，その関係を調査した。調査に使用した入力行列は以下の4つである。これらの入力行列はMatrixMarketとSparse Matrix collection[7]より取得した。

表1:調査に使用した行列

行列名	次元数	非零要素数	解析分野
ex26	2163	94033	流体力学
ex28	2603	77781	流体力学
raefsky2	3242	293551	流体力学
wang1	2903	19093	電気回路

これらの行列をIDR(s)法で e_k の算出に $-E_k c_k - \omega t_k$ を用いた際の $-Aq_k$ との近似誤差とその時点での相対残差ノルムの散布図を図6に示した。パラメータsは各行列について1～30まで動かした。

図6から大部分において近似誤差と相対残差ノルムは比例関係にあると読み取れる。これは相対残差ノルムが大きくなると，$e_k(=r_{k+1}-r_k)$ のノルムも大きくなるため，その時点の e_k から見れば微小であるはずの誤差が収束時には無視できない大きさになるためと考えられる。

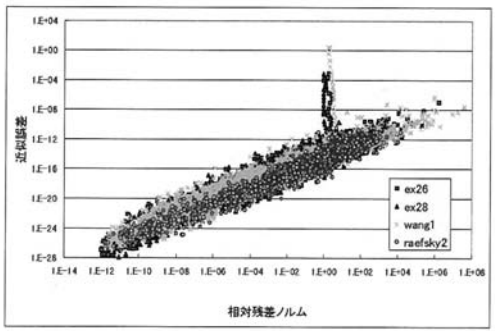

図6:近似誤差と相対残差ノルムの関係

一方で，図6からex26，ex28，wang1の3つの行列について相対残差ノルムが1となる近辺において，近似誤差との相関から大きく外れた点が存在する。これらは最初にKrylov部分空間を生成する反復s回目の点である。この時点はKrylov部分空間内で近似解を探索する前であるため，相対残差ノルムは 10^{-1}～10^{0} 程度である。また，行列raefsky2ではs回目の反復時においても近似誤差がさほど大きな値となっていない。以上から，近似誤差の大きさを示す指標として相対残差ノルム以外のものが必要だと考えられる。

そこで，ex26，ex28，raefsky2につき，s =30，反復30回目における様々な行列，ベクトルを比較したところ，ベクトル c の要素の分布に顕著な差異があるとわかった。ex26，ex28とraefsky2について，s =30，反復30回目におけるベクトル c の要素の絶対値を図7に示す。近似誤差の大きいex26，ex28では最大の要素は 10^{14}，最小の要素が 10^{1} で要素の値の幅が 10^{13} と非常に広い。一方，近似誤差の小さいraefsky2では要素の値の幅は 10^{5} と狭い。

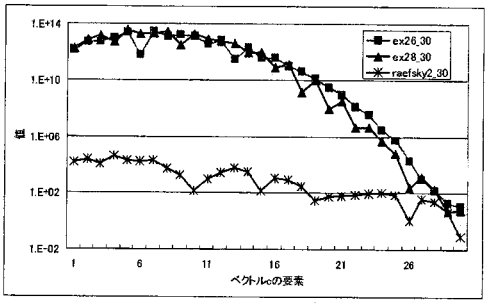

図7:ベクトル c における各要素の値の傾向

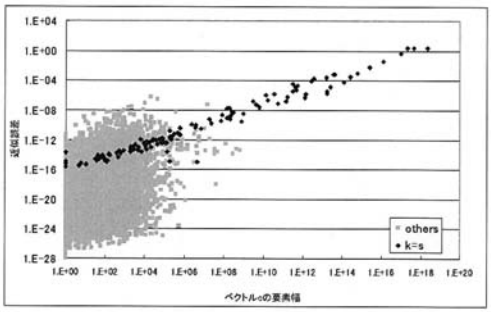

図8:近似誤差とベクトル c の要素幅の関係

そこで，図8に表1の行列，パラメータs＝1～30について近似誤差とベクトル c の要素幅の散布図を示した。図6とは逆に反復s回目の点に強い相関があり，その他の点が無相関となっている。

図6と図8より，s回目の反復時以外で強い相関のある相対残差ノルム $\|r_k\|_2/\|b\|_2$ とs回目の反復時のみに強い相関のあるベクトル c の要素の幅の乗算することで近似誤差に対して常に強い相関を持つようになり，これが近似誤差の大きさを示す指標になると期待できる。図9に相対残差ノルム×[c の要素幅]と近似誤差の散布図を示した。図9

から相対残差ノルム×[c の要素幅]と近似誤差が強い相関を持つことが読み取れる。以上から偽収束の発生を予測する指標 I として相対残差ノルム $\|r_k\|_2/\|b\|_2$ とベクトル c の要素幅の積を用いれば，適切に近似演算と直接演算を使い分けられるとわかった。

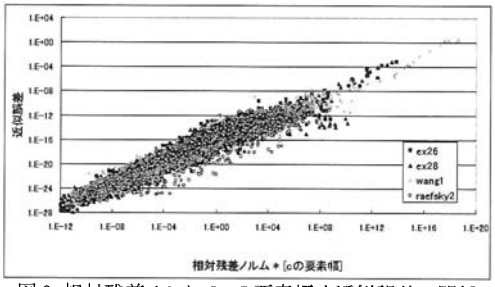

図 9:相対残差ノルム×[c の要素幅]と近似誤差の関係

3.3 Auto Corrected - IDR(s)法の提案

前節で策定した指標 I を用いた実行時自動チューニング方式により高性能と解の信頼性を両立させた Auto Corrected - IDR(s)法(以下，AC-IDR(s)法)を提案する。

従来の IDR(s)法との相違点は，図 1 のアルゴリズムの 16 行において Krylov 部分空間生成後の e_k 算出の際に指標 I が I_{th} 以下ならば近似演算 $-E_k c_k - \omega t_k$，I_{th} 以上ならば直接演算 $-Aq_k$ で求める点である。

直接演算と近似演算のどちらを用いるかの閾値 I_{th} は実行時自動チューニング方式のために新たに設定しなければならないパラメータとなる。しかし，図 9 から I_{th} を要求精度 ε の 10^{11} 倍から 10^{12} 倍に設定すればよいと読み取れるため，I_{th} はライブラリ作成時にあらかじめ値を決めることが可能で，使用者からは見えない内部パラメータにできる。

また，この方式は新たに必要となる演算量とメモリ量が非常に少ないという特徴がある。主な演算は指標 I を求める部分だが，それに必要となる相対残差ノルム $\|r_k\|_2/\|b\|_2$ は 22 行の収束判定時のものを使えばよく，ベクトル c_k の要素幅も s×2 回の大小比較演算と 1 回の除算で求まる。メモリ確保も c_k の最大，最小要素の実数 2 個分である。

このAC-IDR(s)法により，IDR(s)法の利点である高性能を維持しながら偽収束の発生を抑制し，常に要求精度を満たす解を出力することが期待できる。

16: If $I \le I_{th}$ then $e_k = -E_k c_k - \omega t_k$
else $e_k = -Aq_k$
$\left[I = (\|r_k\|_2/\|b\|_2) \times (c_{k\_\max}/c_{k\_\min})\right]$

図10：AC-IDR(s)法のアルゴリズム変更部分

4. 数値実験

4.1 実験条件

提案のAC-IDR(s)法の有効性を確認するため，従来のIDR(s)法と e_k 算出に常に直接演算を用いたDirect-IDR(s)法(以下，D-IDR(s)法と呼ぶ)の2方式に対して演算時間と出力される解の信頼性を比較した。表2に実験環境およびIDR(s)法のパラメータを示す。指標 I の閾値 I_{th} は要求精度εを 10^{-10} としたため，εの 10^{11} 倍となる10と設定した。また，表3に実験に用いた入力行列を示した。これらの行列はMatrixMarketとSparse Matrix collectionより入手した。

表 2 計算機環境とIDR(s)法のパラメータ

CPU	Opteron 2.0GHz
主記憶	16GB(DDR2-667)
s	1～30 pitch 1
要求精度 ε	10^{-10}
閾値 I_{th}	10

表 3 入力行列一覧

行列名	次元数	非零要素数	解析分野
ex26	2163	94033	流体力学
ex28	2603	77781	流体力学
raefsky2	3242	293551	流体力学
wang1	2903	19093	電気回路
wang4	26088	177196	電気回路
memplus	17758	126150	電気回路
poisson3Da	13514	352762	構造解析
Poisson3Db	85623	2374949	構造解析

4.2 実験結果

行列 poisson3Da についてパラメータ s を 1 から 30 まで動かした際の従来の IDR(s)法，D-IDR(s)法，AC-IDR(s)法の 3 種の演算時間を図 11，出力された解の真の相対残差 r_t の 2 ノルムを図 12 に示した。

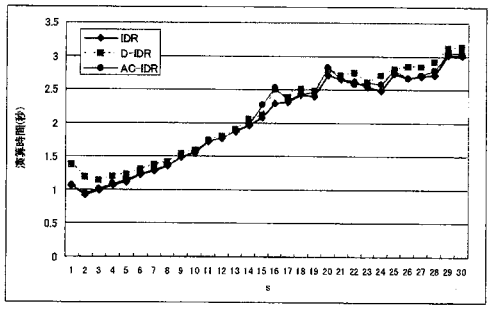

図 11:3 方式の IDR(s)法による poisson3Da の演算時間

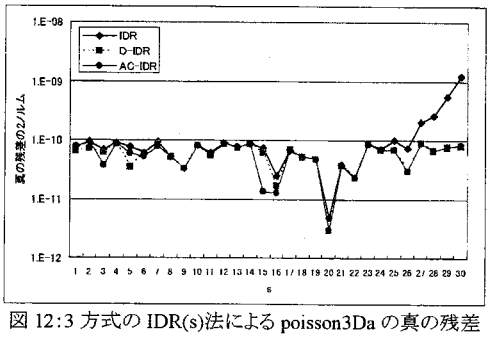

図 12:3 方式の IDR(s)法による poisson3Da の真の残差

図 11 から D-IDR(s)法の演算時間は従来の IDR(s)法と比べて最大で 30%程度増加しているが，AC-IDR(s)法は演算時間増加が D-IDR(s)法と比べて抑制されていることが読み取れる。この演算時間増加の抑制効果は特に s が小さい値のときに顕著で，従来の IDR(s)法と同等の演算時間であった。一方，図 12 から従来の IDR(s)法が s=25 より大きくなると真の相対残差 r_t が要求精度の 10^{-10} を満たさなくなるのに対し，AC-IDR(s)法，D-IDR(s)法は共に全ての s で要求精度を満たしていたと読み取れる。

また，その他の行列の結果を表 4 にまとめた。「演算時間の比率」は従来の IDR(s)法の演算時間を 1 としたときの D-IDR(s)法，AC-IDR(s)法の演算時間の比の s=1～30 での平均値を示し，「解が要求精度を満たした割合」は s=1～30 で真の残差 r_t が要求精度を上回った割合を示す。

表 4 から，AC-IDR(s)法は従来の IDR(s)法から平均で 1.8%演算時間が増加するが，常に要求精度を満たす解を出力していると読み取れる。従来の IDR(s)法は評価した 3 方式の中で平均して最も高い速度が得られたが，要求精度を満たす解を出力している割合は 61%であり，解の信頼性が低かった。D-IDR(s)法は AC-IDR(s)法と同様に常に要求精度を満たす解を出力していたが，従来の IDR(s)法と比較した演算時間の増加は平均で 6.8%であり，演算性能で AC-IDR(s)法に劣っていた。

以上から，提案法の AC-IDR(s)法は常に要求精度を満たす解を出力し，D-IDR(s)法と比べて 4.9%速度向上しており，有効な方式であると確認できた。しかし，従来の IDR(s)法の速度と比較すると 1.7%低下しており，改善の余地があることも確認できた。

5. まとめ

本稿では，近年提案され，その高い演算性能で注目されている IDR(s)法を題材とし，パラメータ s が大きくなると真の相対残差が増加する偽収束が発生し，正しい解が得られない問題の原因究明と解決に取り組んだ。その結果，偽収束発生の原因が残差ベクトルの微小変化ベクトル算出時の近似演算が稀に正しく近似できていないことだと突き止めた。しかし，近似演算を利用しなければ IDR(s)法の高い演算性能が損なわれるという新たな問題が発生する。そこで，偽収束発生を予測する指標を策定し，その指標の値の大小で近似演算と直接演算を切り替える実行時自動チューニング方式を考案した。さらに，その方式を適用することで高い演算性能と解の信頼性を兼ね備えた Auto Corrected - IDR(s)法を提案した。数値実験によると，従来法では要求精度を満たす解を出力した回数が全試行 240 パターンの 61%にあたる 147 パターンであったのに対し，提案法は 240 パターン全てで出力された解が要求精度を満たしていた。また，近似演算を利用しない場合と比べ 4.9%性能向上しており，有効な方式であると確認できた。

また，今回の成果により，自動チューニング方式がソルバの演算時間の高速化・安定化だけではなく，解の信頼性を高める上でも有効であると確認できた。

今後の課題は以下の 2 点である。

1) 本稿で用いた偽収束発生を予測する指標 I は実験的に有効であると確認できたが，理論的な裏付けが取れていない。今後は指標 I の有効性とより良い指標に改良できるかについて分析を進める。
2) 本稿の数値実験では前処理やオーダリングといった高速化・安定化技法を適用せず，並列化も行わなかった。これらの適用時の提案方の有効性について評価する必要がある。

参考文献

[1] 櫻井 隆雄, 直野 健, 恵木 正史, 猪貝 光祥, 木立 啓之; "リスタート付ランチョス法における実行時パラメータ自動チューニング方式の提案", 情報処理学会研究報告 2007-HPC-111, Vol.2007, No.80, pp.173-178 , Aug, 2007.

[2] Sonnenveld, P. and van Gijzen, M. B.; "IDR(s): A family of simple and fast algorithms for solving large nonsymmetric systems of linear equations", TR 07-07, Dept. Math. Anal., Delft, The Netherlands, pp.1-28, 2007.

[3] van der Vorst, H. A.; "Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems, SIAM J. Sci. Stat. Comput, Vol.13, No.2, pp.631-644, 1992.

[4] Y. Saad and M. Schultz; "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems", SIAM J. Sci. Stat. Comput, Vol.7, No.3, pp.856-869, 1986.

[5] 中嶋 徳正, 藤野 清次, 立居場 光生, 尾上 勇介; "多物体電磁波散乱問題の境界要素解析に対する IDR(s)法の性能評価", クリロフ別府フォーラム, pp.47-52, Jan, 2008.

[6] MatrixMarket; "http://math.nist.gov/MatrixMarket/".

[7] Davis, T. A,; Sparse Matrix collection; "http://www.cise.ufl.edu/research/sparse/matrices/".

[8] 土持 秀之, 尾上 勇介, 藤野 清次; "IDR(s)法の有用性の実験的検証 – GMRES(k)との収束性比較 –", 応用数学合同研究集会報告集, pp.260-265, Dec, 2007.

表 4: 各行列における演算時間の比率と解が要求精度を満たした割合

行列	次元数	非零要素数	演算時間の比率			解が要求精度を満たした割合(%)		
			IDR	D-IDR	AC-IDR	IDR	D-IDR	AC-IDR
ex26	2163	94033	1.000	1.101	1.036	56.7	100.0	100.0
ex28	2603	77781	1.000	1.063	0.984	60.0	100.0	100.0
raefsky2	3242	294276	1.000	1.087	1.017	96.7	100.0	100.0
wang1	2903	19093	1.000	1.018	1.000	23.3	100.0	100.0
wang4	26088	177196	1.000	1.070	1.036	53.3	100.0	100.0
memplus	17758	126150	1.000	1.025	0.991	30.0	100.0	100.0
poisson3Da	13514	352762	1.000	1.068	1.019	83.3	100.0	100.0
poisson3Db	85623	2374949	1.000	1.115	1.058	86.7	100.0	100.0
合計値			1.000	1.068	1.018	61.3	100.0	100.0