第10章　連立一次方程式の解法3 — Krylov部分空間法

> 共役勾配法は 1952 年に，M.R.Hestenes と E.Stiefel によって発表された。
> それは，実に華やかなデビューだった。新聞やラジオは「科学史上に残る大発明」として，ビッグ・ニュースの形で報道した。当時，筆者は高校生であったから，詳しい内容は全く分からなかったが
>
> > 連立方程式を解くための画期的な方法が発明された。これまで何百年もの間，n 元の連立方程式を解くには n^3 に比例する演算が必要なもの，とされてきたが，新しい方法によれば，これを n^2 に比例する演算回数で解くことができる … 。
>
> という記事を (中略) はっきり記憶している。数値計算法の一解法が，これほど大々的に報道されたことは，前にも後にも例がない。
>
> 戸川隼人「共役勾配法」(教育出版)

傾斜法は反復法の一種であるが，前章の反復法とは原理的に著しく異なる点がある。そこで新たに 1 章を起こした。

近年よく取り上げられる共役勾配法とその変種は，普通，傾斜法の一種として導出される。ここでは傾斜法の概観を述べた後，その代表的なアルゴリズムである最急降下法と共役勾配法について述べる。

内容

1. **傾斜法 (Gradient Method) —**

   最適化問題のための傾斜法を連立一次方程式に適用できる条件と，その原理を述べる。

2. **最急降下法 (Steepest Desent Method) —**

   最急降下法のアルゴリズムを述べる。

3. **共役勾配法 (Conjugate-Gradient Method) —**

   共役勾配法のアルゴリズムを述べる。

10.1　傾斜法

行列 $A \in M_n(\mathbb{R})$ が正定値対称，即ち任意の $\mathbb{R}^n$ ベクトル $\mathbf{x} \neq 0$ に対し

$$(\mathbf{x}, A\mathbf{x}) = (A\mathbf{x}, \mathbf{x}) \geq 0$$

であるとき，連立一次方程式

$$A\mathbf{x} = \mathbf{b} \tag{10.1}$$

は 2 次形式の極値問題

$$\begin{aligned} f(\mathbf{y}) &= \frac{1}{2}(\mathbf{y}-\mathbf{x}, A(\mathbf{y}-\mathbf{x})) \qquad (10.2) \\ &= \frac{1}{2}(\mathbf{y}-\mathbf{x}, A\mathbf{y}-\mathbf{b}) \\ &= \frac{1}{2}(\mathbf{y}, A\mathbf{y}) - (\mathbf{y}, \mathbf{b}) + \frac{1}{2}(\mathbf{x}, \mathbf{b}) \end{aligned}$$

の解でもある。このとき，極値問題 (10.2) を解くための次のアルゴリズムを，傾斜法という。

アルゴリズム 16 (傾斜法 (Gradient Method))

1. 初期値 $\mathbf{x}_0 \in \mathbb{R}^n$ を決める。
2. $\mathbf{r}_0 := \mathbf{b} - A\mathbf{x}_0$，$\mathbf{p}_0 := \mathbf{r}_0$ とする。
3. $k = 0, 1, 2, \cdots$ に対して以下を計算する。
   (a) $\alpha_k := \dfrac{(\mathbf{r}_k, \mathbf{p}_k)}{(\mathbf{p}_k, A\mathbf{p}_k)}$
   (b) $\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k \mathbf{p}_k$
   (c) $\mathbf{r}_{k+1} := \mathbf{r}_k - \alpha_k A\mathbf{p}_k$(又は $:= \mathbf{b} - A\mathbf{x}_{k+1}$)
   (d) 収束判定
   (e) $\mathbf{p}_{k+1}$ を決める

ここで $\mathbf{r}_k \in \mathbb{R}^n$ を残差ベクトル (residual vector)，$\mathbf{p}_k \in \mathbb{R}^n$ を修正ベクトル (modification vector) という。

残差ベクトル $\mathbf{r}_k$ は

$$\mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k$$

という量であるが，この形だと反復一回毎に $A\mathbf{x}_k$ を計算せねばならず，不経済である。従って普通は

$$\begin{aligned} \mathbf{r}_k &= \mathbf{b} - A\mathbf{x}_k \\ &= \mathbf{b} - A(\mathbf{x}_k + \alpha_k \mathbf{p}_k) \\ &= (\underline{\mathbf{b} - A\mathbf{x}_k}) - \alpha_k A\mathbf{p}_k \\ &= \mathbf{r}_k - \alpha_k A\mathbf{p}_k \end{aligned}$$

という形で再帰的に計算される。

この残差ベクトルの幾何学的意味は次の補題で示される。

補題 10.1.1

$$\nabla f(\mathbf{y}) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\mathbf{y}) \\ \frac{\partial f}{\partial x_2}(\mathbf{y}) \\ \vdots \\ \frac{\partial f}{\partial x_n}(\mathbf{y}) \end{bmatrix} = -(\mathbf{b} - A\mathbf{y}) = -\mathbf{r}$$

(証)

$$\begin{aligned} f(\mathbf{y}) &= \frac{1}{2}(\mathbf{y}, A\mathbf{y}) - (\mathbf{y}, \mathbf{b}) + \frac{1}{2}(\mathbf{x}, \mathbf{b}) \\ &= \frac{1}{2}\sum_{i,j} a_{ij} y_i y_j - \sum_i y_i b_i + \frac{1}{2}(\mathbf{x}, \mathbf{b}) \end{aligned}$$

から

$$\begin{aligned} \frac{\partial f}{\partial y_l}(\mathbf{y}) &= \frac{1}{2}\sum_j a_{lj} y_j + \frac{1}{2}\sum_j a_{jl} y_j - b_l \\ &= \sum_j a_{lj} y_j - b_l \end{aligned}$$

より

$$\nabla f(\mathbf{y}) = -(\mathbf{b} - A\mathbf{x}) = -\mathbf{r}$$

が示された。

(証明終)

この補題より，$\mathbf{x}_k$ に対する残差 $\mathbf{r}_k$ は，点 $\mathbf{x}_k$ における最も勾配のきつい，谷 ($f(\mathbf{y}) = 0$) 方向への傾斜を示していることがわかる。

また α_k はつぎのような性質を持つ。

補題 10.1.2

$$\frac{\partial f}{\partial \alpha_k}(\mathbf{x}_{k+1}) = 0$$

(証)

$$\begin{aligned} f(x_{k+1}) &= \frac{1}{2}(\mathbf{x}_{k+1}, A\mathbf{x}_{k+1}) - (\mathbf{x}_{k+1}, \mathbf{b}) + (\mathbf{x}, \mathbf{b}) \\ &= \frac{1}{2}(\mathbf{x}_k + \alpha_k \mathbf{p}_k, A\mathbf{x}_k + \alpha_k A\mathbf{p}_k) - (\mathbf{x}_k + \alpha_k \mathbf{p}_k, \mathbf{b}) + (\mathbf{x}, \mathbf{b}) \\ &= \frac{1}{2}(\mathbf{x}_k, A\mathbf{x}_k) + \frac{1}{2}(\mathbf{x}_k, \alpha_k A\mathbf{p}_k) + \frac{1}{2}(\alpha_k \mathbf{p}_k, A\mathbf{x}_k) + \frac{1}{2}(\alpha_k \mathbf{p}_k, \alpha_k A\mathbf{p}_k) - (\mathbf{x}_k, \mathbf{b}) - (\alpha_k \mathbf{p}_k, \mathbf{b}) + (\mathbf{x}, \mathbf{b}) \\ &= \frac{1}{2}(\mathbf{x}_k, A\mathbf{x}_k) + \alpha_k(\mathbf{x}_k, A\mathbf{p}_k) + \frac{1}{2}\alpha_k^2(\mathbf{p}_k, A\mathbf{p}_k) - (\mathbf{x}_k, \mathbf{b}) - \alpha_k(\mathbf{p}_k, \mathbf{b}) + (\mathbf{x}, \mathbf{b}) \end{aligned}$$

から

$$
\begin{aligned}
\frac{\partial f}{\partial \alpha_k}(\mathbf{x}_{k+1}) &= (\mathbf{x}_k, A\mathbf{p}_k) + \alpha_k(\mathbf{p}_k, A\mathbf{p}_k) - (\mathbf{p}_k, \mathbf{b}) \\
&= (\mathbf{x}_k, A\mathbf{p}_k) - (\mathbf{p}_k, \mathbf{b}) + (\mathbf{r}_k, \mathbf{p}_k) \\
&= (\mathbf{p}_k, A\mathbf{x}_k) - (\mathbf{p}_k, \mathbf{b}) + (\mathbf{r}_k, \mathbf{p}_K) \\
&= (\mathbf{p}_k, -(\mathbf{b} - A\mathbf{x}_k)) + (\mathbf{r}_k, \mathbf{p}_k) \\
&= -(\mathbf{r}_k, \mathbf{p}_k) + (\mathbf{r}_k, \mathbf{p}_k) \\
&= 0
\end{aligned}
$$

(証明終)

この補題から，α_k が，$\mathbf{p}_k$ という方向において最も極値に近い位置を決めていることがわかる。

傾斜法は，前述のアルゴリズムの修正ベクトル $\mathbf{p}_k$ の決め方によりさまざまな変種を作ることができる。

10.2　最急降下法 (Steepest Decent)

最急降下法は前節の傾斜法において，修正ベクトル $\mathbf{p}_k$ を

$$\mathbf{p}_k := \mathbf{r}_k \tag{10.3}$$

とする。アルゴリズムは次のようになる。

アルゴリズム 17 (最急降下法)

1. 初期値 $\mathbf{x}_0 \in \mathbb{R}^n$ を決める。
2. $\mathbf{r}_0 := \mathbf{b} - A\mathbf{x}_0$ とする。
3. $k = 0, 1, 2, \cdots$ に対して以下を計算する。
   (a) $\alpha_k := \dfrac{\|\mathbf{r}_k\|_2^2}{(\mathbf{r}_k, A\mathbf{r}_k)}$
   (b) $\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k \mathbf{r}_k$
   (c) $\mathbf{r}_{k+1} := \mathbf{r}_k - \alpha_k A\mathbf{r}_k$(又は $:= \mathbf{b} - A\mathbf{x}_{k+1}$)
   (d) $\|\mathbf{x}_{k+1} - \mathbf{x}_k\| < \varepsilon\|\mathbf{x}_k\|$ ならば終了。そうでなければ (a) へ。

このアルゴリズムは基本的には，前節の反復法と数値的な性質は類似点が多く，停止則も同じものが使える。

10.3　共役勾配法

共役勾配法 (Conjugate-Gradient Method, CG 法) のアルゴリズムは以下の通りである。

アルゴリズム 18 (共役勾配法 (Conjugate-Gradient Method))

1. 初期値 $\mathbf{x}_0 \in \mathbb{R}^n$ を決める。

2. $\mathbf{r}_0 := \mathbf{b} - A\mathbf{x}_0$，$\mathbf{p}_0 := \mathbf{r}_0$ とする。
3. $k = 0, 1, 2, \cdots$ に対して以下を計算する。
   (a) $\alpha_k := \dfrac{(\mathbf{r}_k, \mathbf{p}_k)}{(\mathbf{p}_k, A\mathbf{p}_k)}$
   (b) $\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k \mathbf{p}_k$
   (c) $\mathbf{r}_{k+1} := \mathbf{r}_k - \alpha_k A\mathbf{p}_k$(又は $:= \mathbf{b} - A\mathbf{x}_{k+1}$)
   (d) $\beta_k := \frac{\|\mathbf{r}_{k+1}\|_2^2}{\|\mathbf{r}_k\|_2^2}$
   (e) 収束判定
   (f) $\mathbf{p}_{k+1} := \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k$

このアルゴリズムの特徴は

(1) ベクトル計算が多い

(2) 反復法に似てはいるが，有限回の操作で収束する

(3) 数値的性質がよく把握されていない

点にある。(1) は現在流行の並列計算機にとっては都合が良い。1CPU あれば n 回の計算が必要なところを，n CPU ならば 1 回の計算で済む[1]。前章の反復法よりも並列化しやすいのである。この点を踏まえて，共役勾配法系の解法をベクトル反復法という名で分類している本もある。
(2) は次の補題から示される。

補題 10.3.1
共役勾配法において，$0 \leq i < j \leq l \leq n$ のとき

(1) $(\mathbf{r}_j, \mathbf{p}_i) = 0$

(2) $(\mathbf{r}_j, \mathbf{p}_j) = \|\mathbf{r}_j\|_2^2$

(3) $(\mathbf{p}_i, A\mathbf{p}_j) = 0$

(4) $(\mathbf{r}_i, \mathbf{r}_j) = 0$

が成立する。

(証)

ここで $0 \leq i < j \leq k$ とする。$k \leq l$ とし，k に対する帰納法で証明する。まず命題 (A_k) を
(1) $(\mathbf{r}_j, \mathbf{p}_i) = 0$
(2) $(\mathbf{r}_j, \mathbf{p}_j) = \|\mathbf{r}_j\|_2^2$
(3) $(\mathbf{p}_i, A\mathbf{p}_j) = 0$
(4) $(\mathbf{r}_i, \mathbf{r}_j) = 0$

[1]CPU 間の同期をとらなればいけないから，そう単純ではないが。

とする。

(A_0) は (2) が $\mathbf{r}_0 = \mathbf{p}_0$ から自明。それ以外は命題の条件を満たしていない。

(A_k) を仮定すると (A_{k+1}) は次のようにして示される。

(1) $i < j \leq k$ のときは示されているから，まず $i = k$，$j = k + 1$ のときを示せばよい。

$$
\begin{aligned}
(\mathbf{r}_{k+1}, \mathbf{p}_k) &= (\mathbf{r}_k, \mathbf{p}_k) - \alpha_k (A\mathbf{p}_k, \mathbf{p}_k) \\
&= (\mathbf{r}_k, \mathbf{p}_k) - \frac{(\mathbf{r}_k, \mathbf{p}_k)}{(\mathbf{p}_k, A\mathbf{p}_k)} (A\mathbf{p}_k, \mathbf{p}_k) \\
&= 0
\end{aligned}
$$

さらに $(A_k)(1)$，$(A_k)(3)$ から

$$
(\mathbf{r}_{k+1}, \mathbf{p}_j) = (\mathbf{r}_k, \mathbf{p}_j) - \alpha_k (A\mathbf{p}_k, \mathbf{p}_j) = 0
$$

がわかる。

(2) 上の命題を使って

$$
\begin{aligned}
(\mathbf{r}_{k+1}, \mathbf{p}_{k+1}) &= (\mathbf{r}_{k+1}, \mathbf{r}_{k+1}) + \beta_k (\mathbf{r}_{k+1}, \mathbf{p}_k) \\
&= (\mathbf{r}_{k+1}, \mathbf{r}_{k+1})
\end{aligned}
$$

が示された。

(3)

$$
\begin{aligned}
(\mathbf{p}_{k+1}, A\mathbf{p}_j) &= (\mathbf{r}_{k+1}, A\mathbf{p}_j) - \beta_k (\mathbf{p}_k, A\mathbf{p}_j) \\
&= \frac{1}{\alpha_j} (\mathbf{r}_{k+1}, \mathbf{r}_j - \mathbf{r}_{j+1}) - \beta_k (\mathbf{p}_k, A\mathbf{p}_j) \\
&= \frac{1}{\alpha_j} (\mathbf{r}_{k+1}, \mathbf{p}_j - \beta_{j-1}\mathbf{p}_{j-1} - \mathbf{p}_{j+1} + \beta_j \mathbf{p}_j) - \beta_k (A\mathbf{p}_k, \mathbf{p}_k) \\
&= \begin{cases} 0 & \text{for } j < k \quad (A_k)(3) \text{ と } (A_{k+1})(1) \text{ から。} \\ 0 & \text{for } j = k \quad \alpha_k, \beta_k \text{を展開し，} (A_{k+1})(1)(2) \text{ から。} \end{cases}
\end{aligned}
$$

(4) $(A_{k+1})(1)(3)$ から

$$
(\mathbf{r}_j, \mathbf{r}_{k+1}) = (\mathbf{r}_j, \mathbf{r}_k) - \alpha_k (\mathbf{r}_j, A\mathbf{p}_k) = 0。
$$

よって補題は証明された。

(証明終)

この補題の (4) と第 1 章の基本定理から

$$
l \leq n
$$

でなければならない。従って，共役勾配法は n 回以下の反復で真値が得られる。が，実際の数値計算においてはこの条件が満足されず，前章で示した反復法と同様，収束判定に基づいて反復を制御する必要がある。

10.4　共役勾配法の数値的性質

共役勾配法が有限回の代数的操作によって終了するアルゴリズムであることは既に証明した。

しかし，有限桁の浮動小数点数を用いて計算を行うと，大抵の場合，丸め誤差のため真値を得ることができない。そればかりか，限界精度を得ようとすると n 回以上の反復回数を必要とする。

このような，理論と現実との食い違いは割合早くから知られていた。[43] に Ginsburg が寄稿した内容は，もっと以前に同人によって知らされてきたし，[39] にもその記述がある。

共役勾配法の開発チームの一人である T. Ginsburg は [43] に次の言葉を残した。

"$\cdots$ the CG-method has to be treated as if it were an infinite iterative process, proper termination is not trivial."

実際いくつかの問題を解いてみると，限界精度に到達するまでの回数はまちまちである。Ginsburg はその目安として，$3n$〜$5n$ 回まわせばほぼ事足りるとしている。$5n$ 回でも十分でないものは解を得ることはできないであろうとも言っている。

確かに一般的傾向としてなら間違いではない。しかし，反例はいくらでもある。あくまで一つの目安として考えるべきだろう。

例題 10.4.1

$$A = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ \frac{1}{2} & 1 & \frac{2}{3} & \cdots & \frac{2}{n} \\ \frac{1}{3} & \frac{2}{3} & 1 & \cdots & \frac{3}{n} \\ \vdots & \vdots & \vdots & & \vdots \\ \frac{1}{n} & \frac{2}{n} & \frac{3}{n} & \cdots & 1 \end{bmatrix} = \left[\frac{\min(i,j)}{\max(i,j)}\right]$$

を用いて

$$A^T A\mathbf{x} = A^T\mathbf{b}$$

という方程式を解く。次数は $n = 200$ する。ここで真の解は

$$\mathbf{x} = [1\ 1\ \cdots\ 1]^T$$

とした。

これを解くと

反復回数	最大相対誤差
179	1.368×10^{-3}
1879	1.544×10^{-6}

となった。つまり $10n$ 回反復してやっと精度限界までたどり着いたことになる。

この例でもわかるように，問題毎に必要な反復回数は全く異なり，明確な基準を設けることは不可能である。しかも，それは問題の悪条件性にはあまり関係しない [39]。

また，共役勾配法においては停止則の問題は非常に重要である。それは反復法でよく用いられる

$$\|\mathbf{x}_{k+1} - \mathbf{x}_k\| \le \varepsilon \|\mathbf{x}_{k+1}\| \text{ になったら停止する。}$$

という，近似値を直接監視する手法が使えないことである。それは次のような例がめずらしくないことでわかる。

例題 10.4.2

$$A = \begin{bmatrix} n & n-1 & \cdots & 1 \\ n-1 & n & \cdots & 2 \\ \vdots & \vdots & & \vdots \\ 1 & 2 & \cdots & n \end{bmatrix} = [n - |i-j|],\ \mathbf{x} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

とする連立一次方程式

$$A^T A\mathbf{x} = A^T\mathbf{b}$$

を解く。ここで次元数は $n = 20$ とし，IEEE754 倍精度計算を用いた。

$\varepsilon = 10^{-4}$ としたときには，まだ 1〜2 桁しか出ていないのに止まってしまう。

	$\mathbf{x}_4$		$\mathbf{x}_5$	
1	.98681	145741063380E + 00	.98681	276664000060E + 00
2	.10133	414303215180E + 01	.10133	417987338890E + 01
3	.10062	701544218720E + 01	.10062	699858318230E + 01
4	.99568	510096151920E + 00	.99568	519670928390E + 00
5	.99391	548805348430E + 00	.99391	615683767450E + 00
6	.99894	653557090620E + 00	.99894	755009384550E + 00
7	.10037	572541968110E + 01	.10037	582440072840E + 01
8	.10039	092320427600E + 01	.10039	099941670590E + 01
9	.10001	612977915360E + 01	.10001	618390650230E + 01
10	.99655	740790353490E + 00	.99655	783355162320E + 00
11	.99655	740790350030E + 00	.99655	783355158880E + 00
12	.10001	612977914350E + 01	.10001	618390649220E + 01
13	.10039	092320427310E + 01	.10039	099941670300E + 01
14	.10037	572541967950E + 01	.10037	582440072680E + 01
15	.99894	653557082920E + 00	.99894	755009376880E + 00
16	.99391	548805344020E + 00	.99391	615683763070E + 00
17	.99568	510096148570E + 00	.99568	519670925070E + 00
18	.10062	701544218340E + 01	.10062	699858317860E + 01
19	.10133	414303216630E + 01	.10133	417987340350E + 01
20	.98681	145741066940E + 00	.98681	276664003650E + 00

限界精度は

	$\mathbf{x}_2$
1	.10000000000178860E + 01
2	.99999999999774080E + 00
3	.99999999998195110E + 00
4	.99999999998440340E + 00
5	.10000000000041760E + 01
6	.10000000000193850E + 01
7	.10000000000128360E + 01
8	.99999999999306180E + 00
9	.99999999998139700E + 00
10	.99999999998912740E + 00
11	.10000000000090180E + 01
12	.10000000000187760E + 01
13	.10000000000080810E + 01
14	.99999999998862040E + 00
15	.99999999998185020E + 00
16	.99999999999493800E + 00
17	.10000000000136250E + 01
18	.10000000000169100E + 01
19	.10000000000019750E + 01
20	.99999999998447330E + 00

で 12 桁は出る。

では ε を小さくすればよいかというとそうではない。あまり小さくすると，少し条件の悪い問題では止まらないことがある。つまり，この停止則は共役勾配法では下手に使用しないことが望ましい。現在主に用いられているものとしては

$$\|\mathbf{r}_k\| \leq \varepsilon \|\mathbf{r}_0\|$$

となったとき停止するというものである。

最後に，CG 法の計算精度に対する鋭敏性を示す例題を挙げる。

例題 10.4.3
係数行列 A と解 $\mathbf{x}$ を

$$A = \begin{bmatrix} 100 & 99 & \cdots & 1 \\ 99 & 99 & \cdots & 1 \\ \vdots & \vdots & & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix}, \ \mathbf{x} = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 99 \end{bmatrix}$$

とする。この問題に対して CG 法を，IEEE754 単精度 (float), 倍精度 (double)，128bit, 256bit, 512bit, 1024bit でそれぞれ計算し，残差 $\|\mathbf{r}_k\|_2 = \|\mathbf{b} - A\mathbf{x}_k\|_2$ をプロットすると図 10.1 のようになる。

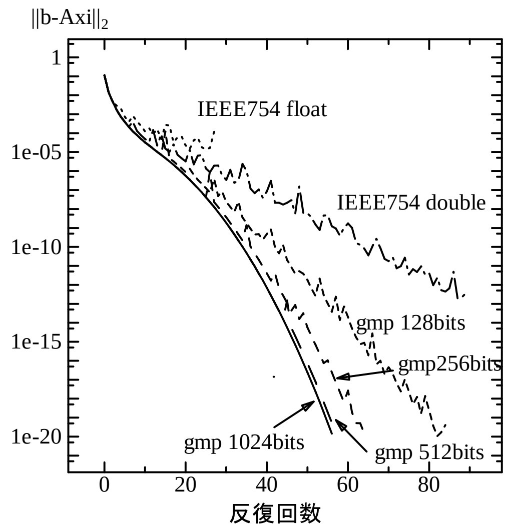

図 10.1: CG 法における丸め誤差の影響

通常は IEEE754 倍精度での数値実験結果のみを使用することが多いが，こうして多倍長計算も行ってみると，如何に CG 法が丸め誤差の影響に敏感であるかが明確となる。特に 512bit 計算でも僅かだが残差のノルムが上昇しており，完全に単調収束させるにはそれ以上の精度が必要であることがわかる。

演習問題

1. CG 法における初期残差 $\mathbf{r}_0$ を元に，係数行列を乗じた $A\mathbf{r}_0, A^2\mathbf{r}_0, ..., A^k\mathbf{r}_0$ によって張られる線型部分空間 K_k を Krylov 部分空間と呼ぶ。この時

$$\mathbf{p}_i \in K_k \ (i = 0, 1, 2, ..., k)$$

であることを示せ[2]。

参考文献

少し古いが，CG 法が考案された歴史的背景なども伺える邦文書として

共役勾配法
戸川隼人
教育出版
1977 年

がある。

[2] これが，CG 法を Krylov 部分空間法と呼んでいる理由である。