高等数值算法与应用（七）

Advanced Numerical Algorithms & Applications

计算机科学与技术系 喻文健

内容概要

- 线性方程组的迭代解法
  - 概述与经典迭代解法回顾
  - 变分原理导出的非经典迭代解法
    - 最速下降法
    - 共扼梯度(CG)算法及预条件技术

$$f(x_k)=\min_{span(p_1\cdots p_k)} f(x) \qquad \|e_k\|_A=\min_{span(p_1\cdots p_k)}\|e\|_A$$

$$\begin{cases} \alpha_k=\dfrac{r_{k-1}^T r_{k-1}}{p_{k-1}^T A p_{k-1}} \\ x_k=x_{k-1}+\alpha_k p_{k-1} \\ r_k=r_{k-1}-\alpha_k A p_{k-1} \\ \beta_k=\dfrac{r_k^T r_k}{r_{k-1}^T r_{k-1}} \\ p_k=r_k+\beta_k p_{k-1} \end{cases}$$

  - Krylov子空间迭代法
    - 投影方法的原理
    - Krylov子空间, Arnodi过程
    - FOM算法, GMRES算法

Yousef Saad's book "*Iterative Methods for Sparse Linear Systems*", 第二版
网络学堂上有电子版

General Projection Method

Wenjian Yu

一般投影方法(projection method)

- 现有的大多数迭代解法都可归类为投影方法，都包含向线性子空间的投影过程
- 基本思想是在$\mathbb{R}^n$的某个子空间(subspace)中寻找方程$Ax=b$的近似解(误差最小, 或残差最小的优化问题)
- 维数为m的子空间$\mathbb{K}$：搜索近似解所使用的子空间

  若准确解x^*与子空间$\mathbb{K}$很接近 …
- 经过推导，上述优化问题常归结为m个正交条件(orthogonality condition)，即要求$b-Ax$垂直于m个线性无关向量 (张成另一子空间$\mathbb{L}$)
- 这些条件被称为Petrov-Galerkin conditions，或Galerkin原理
- 若$\mathbb{L}=\mathbb{K}$，导出的方法称为正交投影(*orthogonal*)方法, 否则为斜投影(*oblique*)方法

  → Galerkin方法

一般投影方法

- $\boldsymbol{V}=(v_1, v_2, \ldots, v_m)$, $\boldsymbol{W}=(w_1, w_2, \ldots, w_m)$, 为$n\times m$列满秩矩阵，记$\mathbb{K}=\text{span}(V)$, $\mathbb{L}=\text{span}(W)$ ，则投影方法为：
  - 寻找 $\boldsymbol{x} \in \mathbb{K}$, 使得 $\boldsymbol{b}-\boldsymbol{Ax} \perp \mathbb{L}$;
  - 若有初始解$\boldsymbol{x}_0$, 则为寻找 $\boldsymbol{x} \in \boldsymbol{x}_0 + \mathbb{K}$, such that $\boldsymbol{b}-\boldsymbol{Ax} \perp \mathbb{L}$;
  - 矩阵形式的方程:

$$\text{Find } x = x_0 + Vy \text{ such that } b - Ax \perp \mathbb{L} \quad \text{leads to}$$

$$(\boldsymbol{b}-\boldsymbol{Ax}_0-\boldsymbol{AVy})^T\boldsymbol{W}=\boldsymbol{0} \Longrightarrow x = x_0 + V(W^TAV)^{-1}W^Tr_0$$

这个过程可以重复执行（每次将上次的解设为$\boldsymbol{x_0}$），直到$\|b-Ax\|$足够小

一般投影方法

ALGORITHM 5.1: Prototype Projection Method

1. *Until convergence, Do:*
2. *Select a pair of subspaces* $\mathbb{K}$ *and* $\mathbb{L}$
3. *Choose bases* $V = [v_1, \ldots, v_m]$ *and* $W = [w_1, \ldots, w_m]$ *for* $\mathbb{K}$ *and* $\mathbb{L}$
4. $r := b - Ax$
5. $y := (W^T AV)^{-1} W^T r$
6. $x := x + Vy$
7. *EndDo*

矩阵 W^TAV 可能奇异

$$W^T AV = \begin{bmatrix} e_1^T A \\ \vdots \\ e_m^T A \end{bmatrix} \begin{bmatrix} e_1 & \cdots & e_m \end{bmatrix}$$

Example 5.2 As an example, consider the matrix

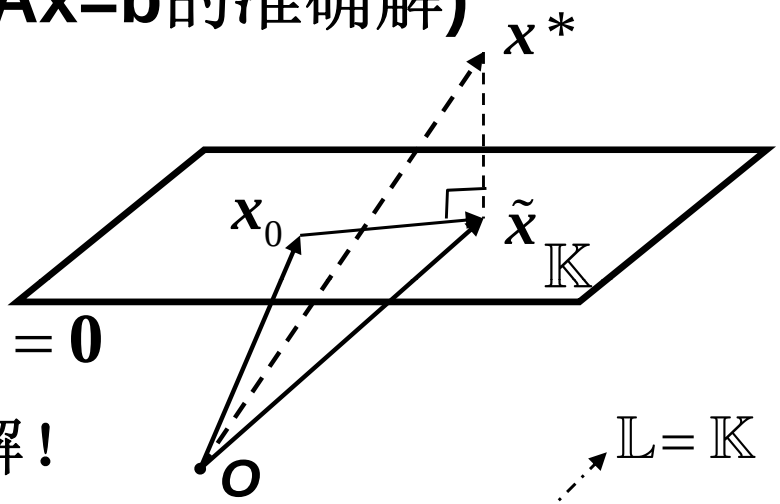
$$A = \begin{pmatrix} O & I \\ I & I \end{pmatrix},$$

$$(W^T AV)_{ij} = e_i^T A e_j = 0 \quad i, j \le m$$

where I is the $m \times m$ identity matrix and O is the $m \times m$ zero matrix, and let $V = W = [e_1, e_2, \ldots, e_m]$. Although A is nonsingular, the matrix $W^T AV$ is precisely the O block in the upper-left corner of A and is therefore singular.

一般投影方法——几何解释(两种优化过程)

- 在$x_0+\mathbb{K}$集合中找 x^*的近似解(x^*为Ax=b的准确解)
- 设$\mathbb{K}$=span(V), V=(v_1, v_2, …, v_m)
- 做法1: $\tilde{x}=x_0+Vy$, $\min\|x^*-\tilde{x}\|$

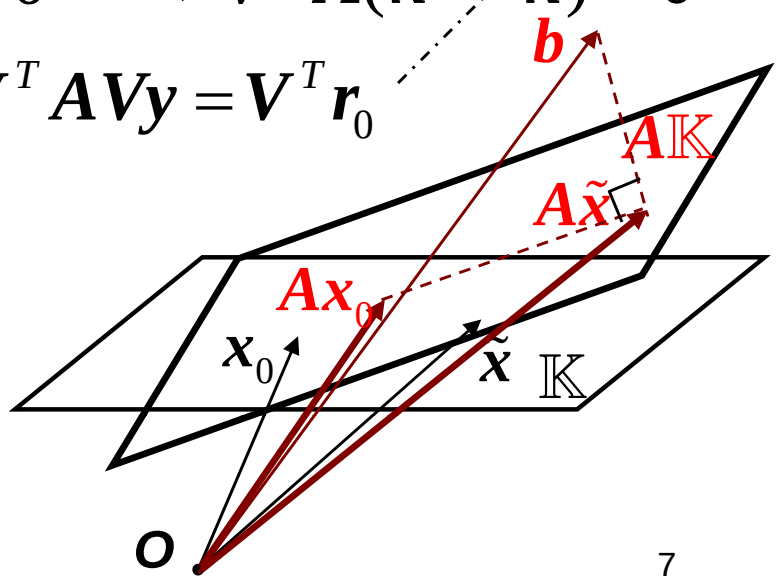


欧式范数: $(x^*-\tilde{x})\perp\mathbb{K}\longrightarrow V^T(x^*-\tilde{x})=0$

$\longrightarrow V^TVy=V^T(x^*-x_0)$ 无法求解!

A范数: $(x^*-\tilde{x})\perp_A\mathbb{K}\longrightarrow(v_i,\ x^*-\tilde{x})_A=0\longrightarrow V^TA(x^*-\tilde{x})=0$

$\longrightarrow V^TAVy=V^TA(x^*-x_0)\longrightarrow V^TAVy=V^Tr_0$



- 做法2: $\tilde{x}=x_0+Vy$, $\min\|b-A\tilde{x}\|$

欧式范数: $b-A\tilde{x}\perp A\mathbb{K}\longrightarrow$

$(AV)^T(b-A\tilde{x})=0\longrightarrow$

$(AV)^TAVy=(AV)^Tr_0$ ⇢ $\mathbb{L}=A\mathbb{K}$

一般投影方法

- 几个理论结论

PROPOSITION 5.1 *Let A, $\mathbb{L}$ and $\mathbb{K}$ satisfy either one of the two following conditions,*

i. *A is positive definite and $\mathbb{L} = \mathbb{K}$, or*

ii. *A is nonsingular and $\mathbb{L} = A\mathbb{K}$*

Then the matrix $B = W^T AV$ is nonsingular for any bases V and W of $\mathbb{K}$ and $\mathbb{L}$ respectively.

Hint: $B=G^TV^TAV$, V^TAV为正定矩阵

$B=(AVG)^TAV=G^T(AV)^TAV$

No breakdown

PROPOSITION 5.2 *Assume that A is Symmetric Positive Definite and $\mathbb{L} = \mathbb{K}$. Then a vector $\tilde{x}$ is the result of an (orthogonal) projection method onto $\mathbb{K}$ with the starting vector x_0 if and only if it minimizes the A-norm of the error over $x_0 + \mathbb{K}$, i.e., if and only if*

$$E(\tilde{x}) = \min_{x \in x_0 + \mathbb{K}} E(x), \quad \text{where } E(x) \equiv \left\| x - x^* \right\|_A^2 = (x - x^*)^T A (x - x^*)$$

与CG法等价

Hint: 设 $x = x_0 + Vy$ 则 $E(x) = y^T V^T A V y - 2 r_0^T V y + \ldots$

$$\min_{x \in x_0 + \mathbb{K}} E(x) \Leftrightarrow \min_{y \in \mathbb{R}^m} (y^T V^T A V y - 2 r_0^T V y) \Leftrightarrow V^T A V y - V^T r_0 = 0$$

即 $V^T (AVy - r_0) = V^T (Ax - b) = 0$，为投影方法的解

一般投影方法

- 几个理论结论(cont'd)

PROPOSITION 5.3 *Let A be an arbitrary square matrix and assume that $\mathbb{L} = A\mathbb{K}$. Then a vector $\tilde{x}$ is the result of an (oblique) projection method onto $\mathbb{K}$ orthogonally to $\mathbb{L}$ with the starting vector x_0 if and only if it minimizes the 2-norm of the residual vector $b - Ax$ over $x \in x_0 + \mathbb{K}$, i.e., if and only if*

$$R(\tilde{x}) = \min_{x \in x_0 + \mathbb{K}} R(x), \quad \text{where } R(x) \equiv \|b - Ax\|_2.$$

GMRES法

Hints: 设 $x = x_0 + Vy$ 则 $R(x) = y^T V^T A^T A V y - 2 r_0^T A V y + \ldots$

$$\min_{x \in x_0 + \mathbb{K}} R(x) \Leftrightarrow \min_{y \in \mathbb{R}^m} (y^T V^T A^T A V y - 2 r_0^T A V y) \Leftrightarrow V^T A^T A V y - V^T A^T r_0 = 0$$

$$\Leftrightarrow V^T A^T (Ax - b) = 0 \Leftrightarrow (b - Ax)^T A V = 0$$

One-dimensional projection processes

$$\mathbb{K} = span\{d\} \quad \text{and} \quad \mathbb{L} = span\{e\}$$

一维子空间

Then $\tilde{x} \leftarrow x + \alpha d$ and Petrov-Galerkin condition $r - \alpha Ad \perp e$ yields

$$\alpha = \frac{(r,e)}{(Ad,e)}$$

(***r*–α*Ad*, *e***) = 0

Three popular choices:

$$\mathbb{K} = \mathbb{L}$$

(I) Steepest descent. A is SPD. Take at each step $d = r$ and $e = r$.

Iteration:

$$\begin{aligned} r &\leftarrow b - Ax, \\ \alpha &\leftarrow (r,r)/(Ar,r) \\ x &\leftarrow x + \alpha r \end{aligned}$$

最小化误差的A范数⇔
最小化 $f(x) = \frac{1}{2}x^T Ax - b^T x$

Each step minimizes $\|x - x^*\|_A^2 = (A(x - x^*), (x - x^*))$

in direction $-\nabla f$. Convergence guaranteed if A is SPD.

(II) Residual norm steepest descent. A is arbitrary (nonsingular). Take at each step $d = A^T r$ and $e = Ad$. $\mathbb{L} = A\mathbb{K}$

$$\alpha = \frac{(r,e)}{(Ad,e)}$$

Iteration:

$$r \leftarrow b - Ax, d = A^T r$$
$$\alpha \leftarrow \|d\|_2^2 / \|Ad\|_2^2$$
$$x \leftarrow x + \alpha d$$

残差/剩余二范数平方函数的梯度

- Each step minimizes $f(x) = \|b - Ax\|_2^2$ in direction $-\nabla f$.

残差/剩余最小

- Important Note: equivalent to usual steepest descent applied to normal equations $A^T Ax = A^T b$.

- Converges under the condition that A is nonsingular.

(III) Minimal residual iteration. A positive definite ($A + A^T$ is SPD).

Take at each step $d = r$ and $e = Ar$. $\mathbb{L} = A\mathbb{K}$

$$\alpha = \frac{(r,e)}{(Ad,e)}$$

Iteration:

$$\begin{aligned} r &\leftarrow b - Ax, \\ \alpha &\leftarrow (Ar, r)/(Ar, Ar) \\ x &\leftarrow x + \alpha r \end{aligned}$$

- Each step minimizes $f(x) = \|b - Ax\|_2^2$ in direction r. 剩余最小
- Converges under the condition that $A + A^T$ is SPD.

Gauss-Seidel方法也可看成是一维投影过程

$\mathbb{K} = \mathbb{L} = span\{e_i\}$

$$\begin{bmatrix} x_1^{(k+1)} \\ \vdots \\ x_{i-1}^{(k+1)} \\ x_i^{(k)} \\ \vdots \\ x_n^{(k)} \end{bmatrix} \longrightarrow \begin{bmatrix} x_1^{(k+1)} \\ \vdots \\ x_{i-1}^{(k+1)} \\ x_i^{(k+1)} \\ \vdots \\ x_n^{(k)} \end{bmatrix}$$

$$X^{(l)} \longrightarrow X^{(l+1)}$$

$$x_i^{(k+1)} = \left(b_i - \sum_{j<i} a_{ij} x_j^{(k+1)} - \sum_{j>i} a_{ij} x_j^{(k)} \right) / a_{ii}$$

$$\left(b - AX^{(l+1)}\right)_i = 0 \Longrightarrow \left(R^{(l+1)}\right)^T e_i = 0$$ 一维投影空间

一维搜索空间： $X^{(l+1)} = X^{(l)} + (x_i^{(k+1)} - x_i^{(k)}) e_i$

Krylov Methods – Arnoldi Procedure

Wenjian Yu

Krylov子空间方法

Choose two spaces of dimension $1 \leq m \leq n$, $\mathbb{K}_m$ and $\mathbb{L}_m$

Petrov-Galerkin methods solve $Ax = b$ by requiring

$$\begin{aligned} x_m &\in x_0 + \mathbb{K}_m \\ b - Ax_m &\perp \mathbb{L}_m \end{aligned}$$

A Krylov method takes

$$\mathbb{K}_m = \mathcal{K}(A, r_0) = span[r_0, Ar_0, \cdots, A^{m-1} r_0]$$

which is a Krylov space.

Krylov子空间方法

Approximations are produced of the form

$$A^{-1}b \approx x_0 + q_{m-1}(A)r_0$$

where q_{m-1} is a polynomial of degree $m-1$.

The methods are strongly related to the choice of the constraint space.

Three cases are of most interest

$$\mathbb{L}_m = \mathbb{K}_m$$ 误差的**A**范数最小**(If A SPD)**

$$\mathbb{L}_m = A\mathbb{K}_m$$ 残差**/**剩余的二范数最小

$$\mathbb{L}_m = \mathcal{K}_m(A^T, r_0)$$ **Lancozs**型

数学上等价的不同算法实现

下面重点考虑一般的非对称**A**矩阵

Arnoldi Procedure

let $A \in \Re^{n\times n}$ be nonsymmetric and consider producing an orthonormal basis for $\mathcal{K}_m(A, v_1)$

给**Krylov**子空间构造单位正交基,
即***V***矩阵: $V_m = [v_1, \cdots, v_m]$

类似于**Gram-Schmidt**正交化过程

v_1 is such that $\|v_1\|_2 = 1$
for $j = 1, \ldots, m$
  $z \leftarrow Av_j$
  Compute $\eta_{ij} = v_i^T z$ for $i = 1, \ldots, j$
  $w_j \leftarrow z - \sum_{i=1}^{j} v_i \eta_{ij}$
  $\eta_{j+1,j} = \|w_j\|_2$
  if $\eta_{j+1,j} = 0$ stop
  $v_{j+1} \leftarrow w_j \eta_{j+1,j}^{-1}$
end for

可用**Arnoldi-MGS**代替

Compute $w_j := Av_j$
For $i = 1, \ldots, j$ *Do:*
  $h_{ij} = (w_j, v_i)$
  $w_j := w_j - h_{ij} v_i$
EndDo

计算量：nm^2次乘法(差不多的加法次数)；
m次矩阵向量乘法

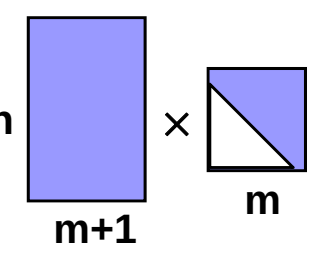
Matrix Form

$$z = \sum_{i=1}^{j} v_i \eta_{ij} + w_j$$

$$A v_j = v_1 \eta_{1j} + v_2 \eta_{2j} + \cdots + v_j \eta_{j,j} + v_{j+1} \eta_{j+1,j}$$

合并 *j*=1,…,*m*的情况

$$A V_m = \overbrace{V_{m+1} \bar{H}_m}$$

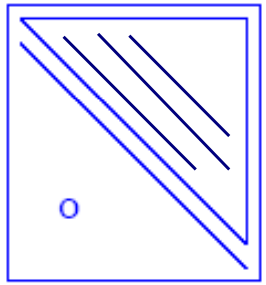


$$= \left(V_m \middle| v_{m+1} \right) \begin{pmatrix} H_m \\ \eta_{m+1,m} e_m^T \end{pmatrix}$$

$$= V_m H_m + w_m e_m^T$$

结果矩阵维度为n×m

扩大的上Hessenberg矩阵 $\bar{H}_m =$



Arnoldi过程每一个循环步生成的矩阵 $\bar{H}_m$ 的一列

H_m为方阵，属于上Hessenberg矩阵

Krylov Methods – Arnoldi (FOM) Algo.

Wenjian Yu

Solving Systems with Arnoldi

Recall our prototype projection method

如果采用正交投影方法

$$\mathbb{K}_m = \mathbb{L}_m = \mathcal{K}_m(A, r_0)$$

x arbitrary;

do until convergence

Select V and W

$\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$

solve $\boldsymbol{M}\boldsymbol{y} = \boldsymbol{W}^T \boldsymbol{r}_0$ with $M = W^T AV$

$x = x + Vy$

end

Construct V_m , **where** $\boldsymbol{v}_1 = \boldsymbol{r}_0 / \|\boldsymbol{r}_0\|$

$\boldsymbol{r}_0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}$

solve $\boldsymbol{M}\boldsymbol{y}_m = \boldsymbol{V}_m^T \boldsymbol{r}_0$ with $M = V_m^T A V_m$

$x = x + V_m y_m$

能否简化计算？

$$\because \quad A V_m = V_m H_m + w_m e_m^T$$

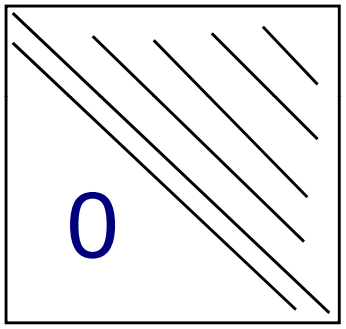
$$\therefore \quad V_m^T A V_m = H_m$$

上Hessenberg矩阵

设 $\beta = \|r_0\|_2$ ，则 $r_0 = v_1 \beta$

$$\therefore \quad V_m^T r_0 = V_m^T v_1 \beta = e_1 \beta$$

需求解的方程 $My_m = V_m^T r_0$ 将形如



$$y_m = \begin{bmatrix} \beta \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$H_m y_m = e_1^{(m)} \beta$$

维度为mxm

问题： 如何确定m的值？

可否在Arnoldi过程中动态确定m？即在子空间维度增大的同时通过剩余判断解是否足够准确，如果准确则停止增加m。

如果这样，希望方便地计算剩余 $\|r_m\|$，而不是依次解 y_m, x_m,然后 $r_m = b - A x_m$

可去掉原型投影方法的最外层循环

不生成x_m的情况下计算剩余的大小

$$
\begin{aligned}
b - Ax_m &= b - A(x_0 + V_m y_m) \\
&= r_0 - AV_m y_m \\
&= v_1\beta - (V_m H_m + v_{m+1} e_m^T \eta_{m+1,m}) y_m \\
&= v_1\beta - V_m \underline{H_m y_m} - v_{m+1} e_m^T y_m \eta_{m+1,m} \\
&= v_1\beta - V_m e_1 \beta - v_{m+1}(e_m^T y_m \eta_{m+1,m}) \\
&= -v_{m+1}(e_m^T y_m)\eta_{m+1,m} \\
\|r_m\|_2 &= |\underline{e_m^T y_m}|\eta_{m+1,m}
\end{aligned}
$$

利用 $\boldsymbol{A}\boldsymbol{V}_m = \boldsymbol{V}_m\boldsymbol{H}_m + \boldsymbol{w}_m\boldsymbol{e}_m^T$
$= \boldsymbol{V}_m\boldsymbol{H}_m + \boldsymbol{v}_{m+1}\eta_{m+1,m}\boldsymbol{e}_m^T$

回顾Arnoldi过程，先有v_m，再计算$\eta_{i,m}$，然后计算w_m(为v_{m+1}准备)
若w_m为零向量，即$\eta_{m+1,m} = 0$，若同时H_m非奇异，则得到准确解;
否则，求解方程得到y_m，通过上公式可算剩余的大小。

新问题：能否推导出从 y_m 得到 y_{m+1} 的简便算法，即利用已有信息加快 $H_{m+1}y_{m+1} = e_1^{(m+1)}\beta$ 的求解？

Incremental Form

$$\left\|r_j\right\|_2 = \left|e_j^T y_j\right| \eta_{j+1,j}$$

求解出y_j的最后一个分量即可计算剩余的大小，

怎么方便地计算y_j的最后一个分量？

通过**Arnoldi**过程使***j*→*j*+1**，如何方便地求出 $\left\|r_{j+1}\right\|_2$ ？

剩余小到满足要求后，如何方便地解出y_m，进而得到x_m?

$$H_j y_j = e_1^{(j)} \beta$$

H_j为上**Hessenberg**矩阵

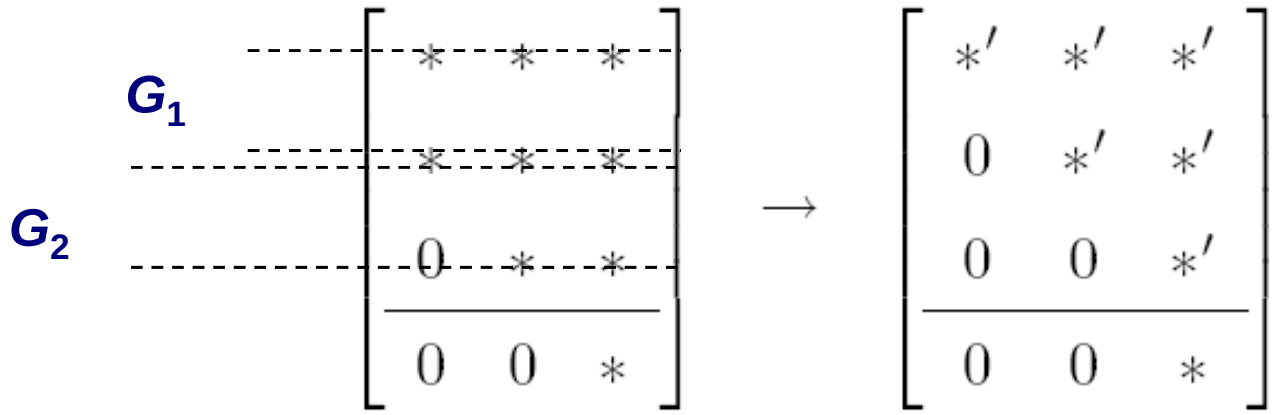
如果方程的系数矩阵是上三角矩阵，则这些问题都迎刃而解！

那我们就用**Givens**旋转将其系数矩阵变为上三角矩阵；

同时在***j*→*j*+1**过程中尽量利用原有信息，得到**Incremental**算法

$j \to j+1$

Consider $j = 3$ $\quad \bar{H}_3 \quad G_2 G_1 \bar{H}_3$



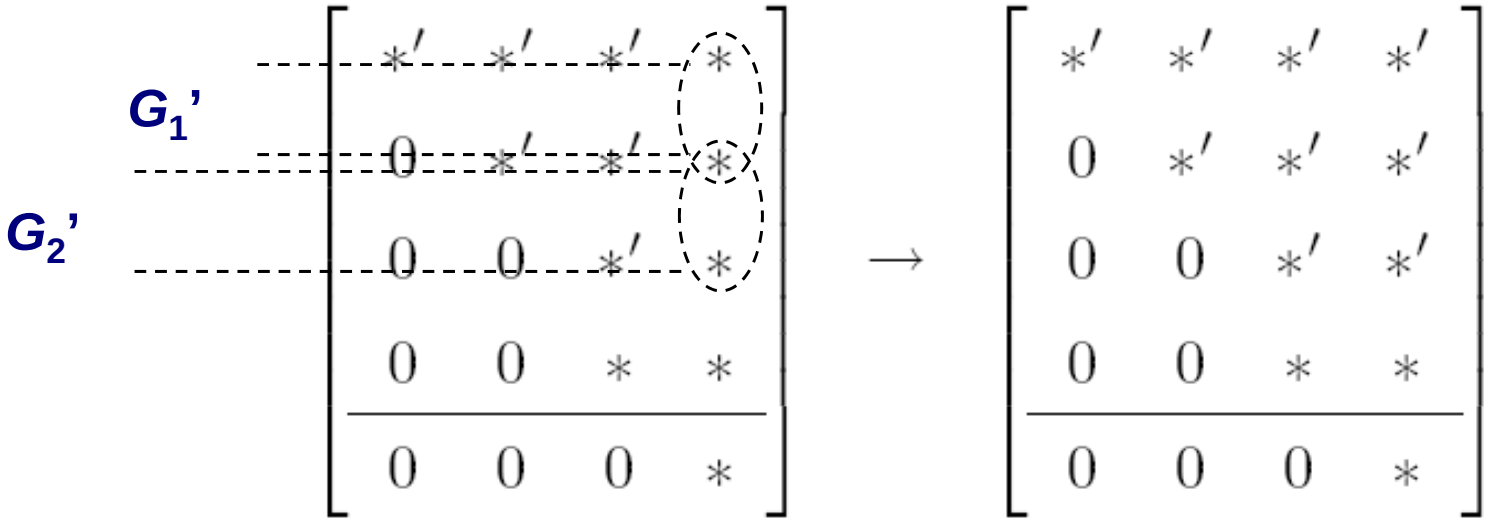
$$H_3 y_3 = e_1^{(3)} \beta$$

$$G_2 G_1 H_3 y_3 = G_2 G_1 e_1^{(3)} \beta$$

很容易得到y_3的最后一个分量

j=4时，算出$\eta_{i,4}$，作为一列扩充矩阵$\bar{H}_4$，然后得$G_2' G_1' \bar{H}_4$

G_1'，G_2' 为与**G_1, G_2**效果相同的**Givens**矩阵，只是维度大一

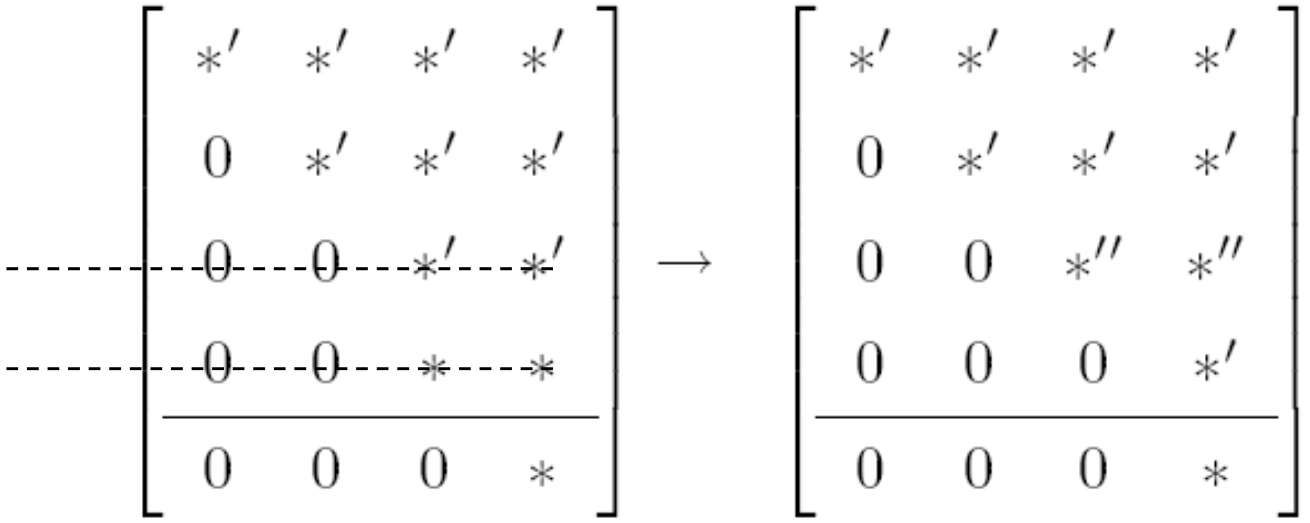


$$G_2' G_1' H_4 y_4 = G_2' G_1' e_1^{(4)} \beta$$

$$j \to j+1$$

$$G_3 G_2' G_1' H_4 y_4 = G_3 G_2' G_1' e_1 \beta \qquad R_4 y_4 = G_3 G_2' G_1' e_1 \beta$$

G_3



很容易得到y_4的最后一个分量 $(y_4)_4$

$$\|r_4\|_2 = |(y_4)_4| \cdot \eta_{5,4}$$

j→j+1的增量操作：Arnoldi过程生成H矩阵新的一列，将以前的Givens旋转作用到这个新列，再构造一个Givens旋转得到新的上三角矩阵和右端项

当剩余大小满足要求时，直接求解上三角方程得到整个y向量，然后利用存储的V矩阵得到解 $x = x_0 + Vy$

Arnoldi算法(全正交算法，Full orthogonalization method, FOM

1. x_0为初始解，$r_0 := b - Ax_0$; $r, \beta := \|r_0\|$;
2. $v_1 := r_0/\beta$; 初始化 H 为空矩阵，d为一维向量; $j := 0$;
3. While $r >$ eps, do　　// Arnoldi-MGS 过程的同时计算剩余的范数
   a) $j := j + 1$, 矩阵 H 扩大一行、一列;
   b) If j=1, $d_j := \beta$, Else 将d扩大一维(新增 0);
   c) $z := Av_j$;
   d) For $i = 1, \ldots, j$,
      $h_{ij} := v_i^T z$; $z := z - h_{ij} v_i$;
   e) EndFor
   f) $w_j := z$; $h_{j+1,j} := \|w_j\|$;
   g) If $h_{j+1,j} = 0$, stop;
   h) $v_{j+1} := w_j / h_{j+1,j}$;
   i) 将$h_{ij}, 1 \le i \le j$填入矩阵H第 j 列;
   j) If $j > 1$,　　//施加 Givens 旋转变换，化H为上三角矩阵
      For $k = 2, \ldots, j-1$,
         施加参数c_{k-1}, s_{k-1}对应的 Givens 旋转变换于矩阵H第 j 列
      EndFor
      计算参数c_{j-1}, s_{j-1}，使其对应 Givens 旋转变换作用于H后使得$H_{j,j-1} = 0$;
      施加参数c_{j-1}, s_{j-1}对应的 Givens 旋转变换于矩阵H第j-1, j 列，以及向量d;
   EndIf
   k) $r := (d_j / H_{j,j}) h_{j+1,j}$;
4. $V := [v_1, v_2, \cdots, v_j]$;
5. 求解上三角线性方程组$Hy = d$;
6. $x := x_0 + Vy$.

r存剩余的二范数
d存右端项βe_1
eps为判断剩余是否足够小的阈值

计算量、存储量

第j次迭代步: 一次矩阵向量乘、j次向量内积、数乘 (n维)

Restarted FOM – FOM(m)

FOM算法的计算量和存储量都随m的增大而增大，并且当m很大时v_j向量的正交性受到舍入误差影响

1. x_0为初始解，$r_0 := b - Ax_0$; $r, \beta := \|r_0\|$;
2. $v_1 := r_0/\beta$; 初始化 H 为空矩阵, d为一维向量; $j := 0$;
3. While $j < m$ && $r >$ eps, do

▪▪▪ ▪▪▪ (同FOM算法)

m为"重启动"阈值

4. $V := [v_1, v_2, \cdots, v_j]$;
5. 求解上三角线性方程组$Hy = d$;
6. $x := x_0 + Vy$.
7. If $r >$ eps, $x_0 := x$, go to 1, EndIf.

致命缺点

对任意非奇异系数阵A，Arnoldi过程可完成，但H_j矩阵奇异，算法执行时发生"恶性中断"！

怎么解决这个问题？

Krylov Methods – GMRES Algo.

Wenjian Yu

Krylov子空间方法

- 一般投影方法的算法

Krylov子空间方法:

$$\mathbb{K}_m = \mathcal{K}_m(A,\ r_0)$$

Do until converge

Select V **and** W **for subspaces** $\mathbb{K}_m$ **and** $\mathbb{L}_m$

$r_0 = b - Ax$

Solve $W^T A V y = W^T r_0$

$x = x + Vy$

End

结合**Arnoldi**过程, 并使 $\mathbb{L}_m = A\mathbb{K}_m$ **(最小剩余法)**

$r_0 = b - Ax$

采用**Arnoldi**过程构造$\mathbb{K}_m$的基向量矩阵 V_m, 其中 $v_1 = r_0 / \|r_0\|$

Solve $W_m^T A V_m y_m = W_m^T r_0$, 其中 $W_m = A V_m$

$x = x + V_m y_m$

如何简化计算?

m该如何取值?

Minimal residual methods ($\mathbb{L}_m = A\mathbb{K}_m$)

When $\mathbb{L}_m = A\mathbb{K}_m$, we let $W_m \equiv AV_m$ and obtain relation

$$W_m^T A V_m y_m = W_m^T r_0$$

$$W_m^T A V_m = (A V_m)^T A V_m$$

$$A V_m = V_{m+1} \bar{H}_m \quad \text{(Arnoldi过程)}$$

$$W_m^T A V_m = \bar{H}_m^T \bar{H}_m$$

$$W_m^T r_0 = (\bar{H}_m^T V_{m+1}^T) r_0 = \bar{H}_m^T \beta e_1$$

and

$$x = x_0 + V_m y_m$$

$$\bar{H}_m^T \bar{H}_m y_m = \bar{H}_m^T \beta e_1$$

即y_m为最小二乘问题 $\bar{H}_m y_m \cong \beta e_1$ 的解!

对一般的非奇异矩阵**A**，不会产生“恶性中断” **(?)**

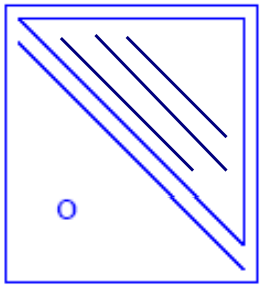
$$\left\| b - A x_m \right\|_2 = \min_{x \in x_0 + \mathbb{K}_m} \left\| b - A x \right\|_2$$

广义最小剩余(残量)法

Generalized Minimal Residual (GMRES) method [1983]

解线性最小二乘问题 $\bar{H}_m y_m \cong \beta e_1$

扩大的上
Hessenberg矩阵



$$y_m \cong \begin{bmatrix} \beta \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

- **问题1：** 如何有效求解？

  QR分解，利用**Givens**旋转方法将 $\bar{H}_m$ 化为上三角

- **问题2：** 如何确定**m**的值？

  可否动态确定**m**？**Arnoldi**过程每做一个循环步，子空间维数就增大一，此时解出相应的 y_m, x_m，判断残差是否足够小，如果足够小则停止增大**m**。

希望方便地计算 $\|r_m\|$，而不是依次解 y_m, x_m，再算 $r_m = b - Ax_m$

可去掉原型投影方法的最外层循环

- Rotation matrices of dimension $m+1$. Define (with $s_i^2 + c_i^2 = 1$):

$$G_i = \begin{pmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & & & \\ & & & c_i & s_i & & \\ & & & -s_i & c_i & & \\ & & & & & 1 & \\ & & & & & & \ddots \\ & & & & & & & 1 \end{pmatrix} \begin{matrix} \\ \\ \\ \leftarrow \text{row } i \\ \leftarrow \text{row } i+1 \\ \\ \\ \\ \end{matrix}$$

- Multiply $\bar{H}_m$ and right-hand side $\bar{g}_0 \equiv \beta e_1$ by a sequence of such matrices from the left. s_i, c_i selected to eliminate $h_{i+1,i}$

Example:

$$\bar{H}_5 = \begin{pmatrix} h_{11} & h_{12} & h_{13} & h_{14} & h_{15} \\ h_{21} & h_{22} & h_{23} & h_{24} & h_{25} \\ & h_{32} & h_{33} & h_{34} & h_{35} \\ & & h_{43} & h_{44} & h_{45} \\ & & & h_{54} & h_{55} \\ & & & & h_{65} \end{pmatrix}, \quad \bar{g}_0 = \begin{pmatrix} \beta \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

- **1-st Rotation**

$$G_1 = \begin{pmatrix} c_1 & s_1 & & & & \\ -s_1 & c_1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{pmatrix}$$

with

$$s_1 = \frac{h_{21}}{\sqrt{h_{11}^2 + h_{21}^2}}, \quad c_1 = \frac{h_{11}}{\sqrt{h_{11}^2 + h_{21}^2}}$$

$$\bar{H}_m^{(1)} = \begin{pmatrix} h_{11}^{(1)} & h_{12}^{(1)} & h_{13}^{(1)} & h_{14}^{(1)} & h_{15}^{(1)} \\ & h_{22}^{(1)} & h_{23}^{(1)} & h_{24}^{(1)} & h_{25}^{(1)} \\ & h_{32} & h_{33} & h_{34} & h_{35} \\ & & h_{43} & h_{44} & h_{45} \\ & & & h_{54} & h_{55} \\ & & & & h_{65} \end{pmatrix}, \quad \bar{g}_1 = \begin{pmatrix} c_1\beta \\ -s_1\beta \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

- repeat with $G_2, \ldots, G_i$. Result:

$$\bar{H}_5^{(5)} = \begin{pmatrix} h_{11}^{(5)} & h_{12}^{(5)} & h_{13}^{(5)} & h_{14}^{(5)} & h_{15}^{(5)} \\ & h_{22}^{(5)} & h_{23}^{(5)} & h_{24}^{(5)} & h_{25}^{(5)} \\ & & h_{33}^{(5)} & h_{34}^{(5)} & h_{35}^{(5)} \\ & & & h_{44}^{(5)} & h_{45}^{(5)} \\ & & & & h_{55}^{(5)} \\ & & & & 0 \end{pmatrix}, \quad \bar{g}_5 = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ . \\ . \\ \gamma_6 \end{pmatrix}.$$

实际的算法中，这是增量式(incremental)过程

经过执行**Givens**变换，最小二乘问题变为 $\bar{H}_m^{(m)} y_m \cong \bar{g}_m$

记 $\bar{H}_m^{(m)}$和$\bar{g}_m$的前**m**行分别为R_m和g_m，则得方程

$$R_m y_m = g_m$$ (系数矩阵为上三角方阵)

原始方程的剩余

$$\begin{aligned} b - Ax_m &= b - Ax_0 - AV_m y_m \\ &= r_0 - V_{m+1}\bar{H}_m y_m \\ &= v_1\beta - V_{m+1}\bar{H}_m y_m \\ &= V_{m+1}e_1\beta - V_{m+1}\bar{H}_m y_m \\ &= V_{m+1}(\beta e_1 - \bar{H}_m y_m) \end{aligned}$$

所以，

$$\|b - Ax_m\|_2 = \|\beta e_1 - \bar{H}_m y_m\|_2$$

$$\overset{?}{=} |\gamma_{m+1}|$$

记 $\bar{g}_m = (\gamma_1, \ldots, \gamma_m, \gamma_{m+1})^T$

若**Arnoldi**过程中断，意味着什么？

$h_{m+1,m} = 0$ (不影响$AV_m = V_{m+1}\bar{H}_m$) ⇒ 不需要最后一次旋转，

$\gamma_{m+1} = 0$，因此剩余为**0**！

GMRES(m)算法

1. x_0为初始解，$r_0 := b - Ax_0$; $\gamma, \beta := \|r_0\|$;
2. $v_1 := r_0/\beta$; 初始化g为长度为 m 的一维数组;
3. $g[1] = \beta$; $j := 0$;
4. While $j < m$ && $\gamma >$ eps, do　　// Arnoldi-MGS 过程的同时计算剩余的范数
   a) $j := j + 1$;
   b) If j==1, $\bar{H}$为 2×1 矩阵; Else 将$\bar{H}$扩大一行、一列;
   c) $z := Av_j$;
   d) For $i = 1, \ldots, j$,
      $h_{ij} := v_i^T z$; $z := z - h_{ij} v_i$;
   e) EndFor
   f) $w_j := z$; $h_{j+1,j} := \|w_j\|$;
   g) If $h_{j+1,j} != 0$, $v_{j+1} := w_j/h_{j+1,j}$;
   h) 将$h_{ij}, 1 \le i \le j+1$, 填入矩阵$\bar{H}$的第 j 列;
   i) If $j > 0$,　　//施加 Givens 旋转变换，化$\bar{H}$为上三角矩阵
      For $k = 2, \ldots, j$,
         施加参数c_{k-1}, s_{k-1}对应的 Givens 旋转变换于矩阵$\bar{H}$第 j 列;
      EndFor
      计算参数c_j, s_j，使其对应 Givens 旋转变换作用于$\bar{H}$后使得$\bar{H}_{j+1,j} = 0$;
      施加参数c_j, s_j对应的 Givens 旋转变换于矩阵$\bar{H}$第 j 列，以及向量g;
      EndIf
   j) $\gamma := |g[j+1]|$;
5. $V := [v_1, v_2, \cdots, v_j]$;
6. 求解上三角线性方程组$\bar{H}[1:j, :]y = g[1:j]$;
7. $x := x_0 + Vy$;
8. If $\gamma > eps$, $x_0 := x$, go to 1, EndIf.

m为"重启动"阈值

γ为剩余向量的二范数;
g为最小二乘问题右端项;
eps为判断剩余是否足够小的阈值.

计算量、存储量(同FOM算法)

第j次迭代步: 一次矩阵向量乘、j次向量内积、数乘 (n维)

与FOM算法一样，GMRES的计算量和存储量都随迭代步数的增大而增大，并且，当j很大时v_j向量的正交性受到舍入误差影响

解决办法1：“重启动”的GMRES，即GMRES(m)算法 (见前页). m为”重启动”阈值；将投影法最外层循环保留.

解决办法2：截断Arnoldi过程中的正交化过程 (Incomplete orthogonalization)

办法3：通过预条件技术，改善待求解问题系数矩阵的性态，加速GMRES算法的收敛过程。

关于收敛性：

1.若无舍入误差， GMRES(∞)算法在n步一定收敛。

2.若无舍入误差，且A正定，则GMRES(m)算法收敛。

3. 对一般矩阵A，GMRES(m)算法并不保证收敛，剩余的减小可能停滞

ALGORITHM : GMRES – No preconditioning

1. **Start:** *Choose* x_0 *and a dimension* m *of the Krylov subspaces.*

2. **Arnoldi process:**

   - *Compute* $r_0 = b - Ax_0$, $\beta = \|r_0\|_2$ *and* $v_1 = r_0/\beta$.
   - *For* $j = 1, \ldots, m$ *do*
     - *Compute* $w := Av_j$
     - *for* $i = 1, \ldots, j$, *do* $\left\{ \begin{array}{l} h_{i,j} := (w, v_i) \\ w := w - h_{i,j} v_i \end{array} \right\}$;
     - $h_{j+1,1} = \|w\|_2$; $v_{j+1} = \frac{w}{h_{j+1,1}}$
   - *Define* $V_m := [v_1, \ldots, v_m]$ *and* $\bar{H}_m = \{h_{i,j}\}$.

3. **Form the approximate solution:** *Compute* $\boxed{x_m = x_0 + V_m y_m}$ *where* $y_m = \mathrm{argmin}_y \|\beta e_1 - \bar{H}_m y\|_2$ *and* $e_1 = [1, 0, \ldots, 0]^T$.

4. **Restart:** *If satisfied stop, else set* $x_0 \leftarrow x_m$ *and goto 2.*

ALGORITHM : **GMRES – with (right) Preconditioning**

$AM^{-1}u = b$, with $x = M^{-1}u$

1. Start: *Choose* x_0 *and a dimension* m *of the Krylov subspaces.*

2. Arnoldi process:

- *Compute* $r_0 = b - Ax_0$, $\beta = \|r_0\|_2$ *and* $v_1 = r_0/\beta$.
- *For* $j = 1, \ldots, m$ *do*
  - *Compute* $z_j := M^{-1}v_j$
  - *Compute* $w := Az_j$
  - *for* $i = 1, \ldots, j$, *do :* $\left\{ \begin{array}{l} h_{i,j} := (w, v_i) \\ w := w - h_{i,j}v_i \end{array} \right\}$
  - $h_{j+1,1} = \|w\|_2$; $v_{j+1} = w/h_{j+1,1}$
- *De£ne* $V_m := [v_1, \ldots, v_m]$ *and* $\bar{H}_m = \{h_{i,j}\}$.

3. Form the approximate solution: *Compute* $\boxed{x_m = x_0 + M^{-1}V_m y_m}$

where $y_m = \mathrm{argmin}_y \|\beta e_1 - \bar{H}_m y\|_2$ *and* $e_1 = [1, 0, \ldots, 0]^T$.

4. Restart: *If satis£ed stop, else set* $x_0 \leftarrow x_m$ *and goto 2.*

优于左预条件之处：

1. 算原问题的剩余

2. 易于构造**Flexible**预条件

ALGORITHM : GMRES – with variable Preconditioning

1. Start: *Choose* x_0 *and a dimension* m *of the Krylov subspaces.*

2. Arnoldi process:

Flexible GMRES

- *Compute* $r_0 = b - Ax_0$, $\beta = \|r_0\|_2$ *and* $v_1 = r_0/\beta$.
- *For* $j = 1, ..., m$ *do*
  - *Compute* $z_j := M_j^{-1} v_j$; *Compute* $w := Az_j$;
  - *for* $i = 1, \ldots, j$, *do:* $\left\{ \begin{array}{l} h_{i,j} := (w, v_i) \\ w := w - h_{i,j} v_i \end{array} \right\}$;
  - $h_{j+1,1} = \|w\|_2; v_{j+1} = w/h_{j+1,1}$
- *Define* $Z_m := [z_1, ..., z_m]$ *and* $\bar{H}_m = \{h_{i,j}\}$.

3. Form the approximate solution: *Compute* $\boxed{x_m = x_0 + Z_m y_m}$ *where*

$y_m = \text{argmin}_y \|\beta e_1 - \bar{H}_m y\|_2$ *and* $e_1 = [1, 0, \ldots, 0]^T$.

4. Restart: *If satisfied stop, else set* $x_0 \leftarrow x_m$ *and goto 2.*

Properties

- x_m minimizes $b - Ax_m$ over $\mathrm{Span}\{Z_m\}$.
- If $Az_j = v_j$ (i.e., if preconditioning is 'exact' at step j) then approximation x_j is exact.
- If M_j is constant then method is $\equiv$ to Right-Preconditioned GMRES.

Additional Costs:

见Saad的书， §9.4.1

- Arithmetic: none.
- Memory: Must save the additional set of vectors $\{z_j\}_{j=1,\ldots m}$

Advantage: Flexibility

小结

- 基于Arnoldi过程的方法
  - FOM（$\mathbb{L}_m=\mathbb{K}_m=\mathcal{K}_m(\boldsymbol{A}, \boldsymbol{r_0})$，解方程 $H_m y_m = \beta e_1$，恶性中断）
  - GMRES（$\mathbb{L}_m= \boldsymbol{A}\mathbb{K}_m$，解最小二乘问题 $\bar{H}_m y_m \cong \beta e_1$ ）
- 实用技术
  - Arnldi-MGS过程
  - 基于Givens旋转的incremental方法，剩余计算
- 改善迭代步增大时计算效率的技术
  - Restart: GMRES(m), 截断正交化
  - Precondition