MATH20007h.01 数值算法与案例分析 (H) Fall 2025

Lecture 11 稀疏矩阵迭代法 2 — 2025.11.24

教授：邵美悦

Scribe: 路人甲

1 大纲

1. Krylov 子空间方法
2. 最速下降法
3. 混合精度算法 (H)

2 Krylov 子空间方法

2.1 GMRES 方法

我们考虑线性方程组 $Ax = b$，其中 $A \in \mathbb{C}^{n\times n}$ 非奇异——并且可能非常稀疏。它的解无需多说，自然是 $x = A^{-1}b$. 这显然不是一个很好的选择——其一，计算 A^{-1} 的开销很大，这一步的操作数就是 $O(n^3)$，显然不适合用于大规模矩阵计算；其二，即使 A 很稀疏，A^{-1} 往往是一个稠密矩阵，一方面这使得存储开销大增，另一方面也使得矩阵–向量乘法的计算开销大增。那么，我们有没有什么方法能够规避这几个问题呢？

问题的症结在于 A^{-1} 可能很稠密——但我们能不能把 A^{-1} 表示成 A 的多项式呢？如果可以的话，那么我们就可以通过多次矩阵–向量乘法来计算 $A^{-1}b$，而这些矩阵–向量乘法本身的开销并不大，因为 A 很稀疏。这样，我们就可以避免计算和存储 A^{-1}，从而节省计算和存储开销。那么这个操作是否可行呢？我们在高等线性代数课上都学过 Cayley–Hamilton 定理。它告诉我们，若 A 的特征多项式是 $p(\lambda)$，那么我们就有矩阵函数恒等式——$p(A) = 0$. 不妨设 $p(A) = \alpha_n A^n + \cdots + \alpha_1 A + \alpha_0$，由于 A 非奇异，我们有

$$\alpha_n A^{n-1} + \cdots + \alpha_1 I + \alpha_0 A^{-1} = 0,$$

亦即

$$A^{-1} = -\frac{1}{\alpha_0}\left(\alpha_n A^{n-1} + \cdots + \alpha_1 I\right).$$

当然，一切的前提是 $\alpha_0 \neq 0$，也就是说 0 不能是 A 的特征值——这是自然的，因为我们已经假设 A 非奇异了。于是，我们就可以通过计算 A 的特征多项式，来间接地表示 A^{-1}.

但是，计算特征多项式本身并不是一件容易的事——特征值算法里最快最稳定的 QR 算法都要 $O(n^3)$ 的时间复杂度，可见这个问题和计算 A^{-1} 本身一样困难。既然完整的特征多项式很难算，那么我们能不能用低阶多项式去近似 A^{-1} 呢？我们不妨考虑如下迭代过程：

$$x_k \in \mathrm{span}\{b, Ab, A^2b, \ldots, A^{k-1}b\},$$

其中 x_k 是迭代的第 k 步近似解。我们把由向量集合 $\{b, Ab, A^2b, \ldots, A^{k-1}b\}$ 所张成的子空间称为 A 关于 b 的第 k 个 Krylov 子空间，记作 $\mathcal{K}_k(b)$. 那么，我们知道 x_n 是一定准确的。现在我们的目标就是，在保证精度的前提下，尽可能让 k 小一点。那么，我们该如何选择 x_k 呢？一种自然的选择是使得**残差最小化**的 x_k，即

$$\min_{x_k \in \mathcal{K}_k(b)} \|b - Ax_k\|_2.$$

现在我们考虑另一种在数学上完全等价的迭代方式。如果我们记 $r_0 = b - Ax_0$，考虑 $A\Delta x = r_0$ 的迭代过程

$$\Delta x_k \in \mathcal{K}_k(r_0),$$

并令 $x_k = x_0 + \Delta x_k$. 那么，我们有

$$b - Ax_k = b - A(x_0 + \Delta x_k) = r_0 - A\Delta x_k.$$

因此，选择使得 $\|b - Ax_k\|_2$ 最小化的 x_k，等价于选择使得 $\|r_0 - A\Delta x_k\|_2$ 最小化的 Δx_k. 这两种迭代方式在数学上是等价的，但在实际计算中可能会有不同的数值表现——一般来说后者会更好一些，因为它避免了在每一步迭代中都计算 Ax_k，从而减少了数值误差的积累。当然，后者的优势主要体现在初值 x_0 比较好的情况下。若初值比较坏，二者大差不差。

不管采用哪种策略，我们都可以把问题归结为如何求解一个带约束的最小二乘问题：

$$\min_{\Delta x_k \in \mathcal{K}_k(r_0)} \|r_0 - A\Delta x_k\|_2.$$

最小二乘问题与 QR 分解密切相关。因此，我们不妨考虑找子空间 $\mathcal{K}(b)$ 的一组正交基 Q_k，并设 $\Delta x_k = Q_k y_k$. 那么，我们只需求解

$$\min_y \|r_0 - AQ_k y\|_2.$$

这就变成了一个无约束的最小二乘问题了——这类问题我们肯定会解。但是，求解这个最小二乘问题仍然很困难——矩阵的维数可能很大——我们是否能利用 Krylov 子空间的特殊结构来简化计算呢？我们知道，Krylov 子空间的正交化依赖于 Arnoldi 过程——这是我们所熟知的结论。现在，我们就来分析一下 Arnoldi 过程在这里的作用。如果设 $Q_k = [q_1, \ldots, q_k]$，那么我们有

$$A\begin{bmatrix} q_1 & q_2 & \cdots & q_k \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & \cdots & q_{k+1} \end{bmatrix} \begin{bmatrix} \times & \times & \cdots & \times \\ \times & \times & \cdots & \times \\ & \times & \cdots & \times \\ & & \ddots & \vdots \\ & & & \times \end{bmatrix}_{(k+1)\times k}.$$

这里我们得到了一个上 Hessenberg 矩阵，我们不妨记作 $\tilde{H}_k$，于是

$$AQ_k = Q_{k+1}\tilde{H}_k.$$

因此，我们有

$$\begin{aligned}\|r_0 - AQ_k y_k\|_2 &= \|r_0 - Q_{k+1}\tilde{H}_k y_k\|_2 \\ &= \|Q_n^\star r_0 - Q_n^\star Q_{k+1}\tilde{H}_k y_k\|_2 \quad \text{2-范数的酉不变性}\end{aligned}$$

由于 Q_n 的第 1 列与 r_0 平行而第 2 列到第 n 列都与 r_0 正交，因此 $Q_n^\star r_0 = e_1\|r_0\|_2$，其中 $e_1 \in \mathbb{R}^n$. 另一方面，

$$Q_n^\star Q_{k+1} = \begin{bmatrix} I_{k+1} \\ 0 \end{bmatrix}$$

因此

$$\begin{aligned}\|e_1\|r_0\|_2 - Q_n^\star Q_{k+1}\tilde{H}_k y_k\|_2 &= \left\| e_1\|r_0\|_2 - \begin{bmatrix} I_{k+1} \\ 0 \end{bmatrix} \tilde{H}_k y_k \right\|_2, \quad e_1 \in \mathbb{R}^n \\ &= \|e_1\|r_0\|_2 - \tilde{H}_k y_k\|_2, \quad e_1, y_k \in \mathbb{R}^{k+1}\end{aligned}$$

想必读者也能看到这一系列变换当中的玄机了——原来的 n 维最小二乘问题被化简成了一个 $(k+1)$ 维的最小二乘问题，而 k 往往远小于 n. 这就是 Krylov 子空间方法的精髓所在。而对于这个低维的最小二乘问题，我们完全可以使用传统的 QR 分解方法来求解。这里宜使用 Givens 旋转来进行 QR 分解，因为矩阵已经非常接近上三角了。具体地（我们以 $k = 5$ 为例）

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \\ & & & & \times \end{bmatrix} y_k = \begin{bmatrix} \times \\ \times \\ \times \\ \times \\ \times \\ \times \end{bmatrix}$$

经过 Givens 旋转后，系数矩阵变成上三角矩阵，右端项也相应地修正。于是

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \\ & & & & \times \\ & & & & 0 \end{bmatrix} y_k = \begin{bmatrix} \times \\ \times \\ \times \\ \times \\ \times \\ \times \end{bmatrix}.$$

现在我们只需要解上三角方程组的前 k 个方程就可以了，第 $k+1$ 个方程我们管不了了——这正是方程的残量。我们依据余项的大小来确定是否停机。这种方法被称为 GMRES 方法 (Generalized Minimum Residual Method)。

现在我们来对 GMRES 方法的收敛性做一些分析。首先，残量是单调下降至零的。这是因为 Krylov 子空间是单调扩张的，因此

$$\|r_{k+1}\|_2 = \min_{\Delta x_{k+1}\in\mathcal{K}_{k+1}(r_0)} \|r_0 - A\Delta x_{k+1}\|_2 \leq \min_{\Delta x_k\in\mathcal{K}_k(r_0)} \|r_0 - A\Delta x_k\|_2 = \|r_k\|_2.$$

但是，残量大小并不是严格单调下降的——完全有可能前 $n-1$ 步残量接近恒定，直到第 n 步骤才骤然下降为零。例如 Frobenius 友阵

$$\begin{bmatrix} 0 & 0 & \cdots & 1 \\ 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 1 & 0 \end{bmatrix} x = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

我们一眼就能看出它的解是 $x = [0, 0, \cdots, 1]^{\mathrm{T}}$，但是如果我们使用 GMRES 方法的话，每一步的残量都原封不动地向下传递，直到第 n 步才会骤然变成零。事实上，有理论证明，任意给定残量下降曲线与特征值或奇异值情况等，我们都能够构造出一个矩阵，使得 GMRES 方法的残量恰好按照给定的曲线下降。因此，GMRES 方法的收敛性分析仍然是一个开放性的问题。

2.2 FOM 方法

同属于 Krylov 子空间方法的还有 FOM 方法 (Full Orthogonalization Method)。它的核心思想是：我们要让残量与 Krylov 子空间正交。也就是说，我们要求 $r_k \perp Q_k$，这被称为 Galerkin 条件。接下来，我们就来推导 FOM 方法的迭代格式。正交化条件要求

$$Q_k^\star(r_0 - A\Delta x_k) = 0, \quad \Delta x_k \in \mathcal{K}_k(r_0) = \operatorname{span} Q_k.$$

由于 $\Delta x_k \in \operatorname{span} Q_k$，我们设 $\Delta x_k = Q_k y_k$. 于是上式等价于

$$Q_k^\star r_0 - Q_k^\star A Q_k y_k = 0.$$

而在前面的分析中我们已经得到 $Q_k^\star r_0 = e_1\|r_0\|_2$；另一方面，由 Arnoldi 过程我们也知道 $AQ_k = Q_{k+1}\tilde{H}_k$，因此 $Q_k^\star A Q_k = Q_k^\star Q_{k+1}\tilde{H}_k$，而

$$Q_k^\star Q_{k+1} = \begin{bmatrix} I_k & 0 \end{bmatrix}.$$

因此，$Q_k^\star Q_{k+1}\tilde{H}_k = [I_k, 0]\begin{bmatrix} H_k \\ c \end{bmatrix} = H_k$.

综上所述，FOM 方法要求解如下线性方程组：

$$H_k y_k = e_1\|r_0\|_2.$$

因此迭代格式为

$$x_k = x_0 + Q_k y_k, \quad \text{其中 } H_k y_k = e_1 \|r_0\|_2.$$

从另一个角度看，FOM 方法实际上是 GMRES 方法的简化。我们知道 Arnoldi 过程满足 $AQ_k = Q_{k+1}\tilde{H}_{k+1} = Q_k H_k + \text{rank1} \approx Q_k H_k$，其中 rank1 表示秩一修正项。因此，这种方法在直观上是非常好理解的。

如果说，FOM 方法是让残量与 Krylov 子空间正交的话，那么 GMRES 方法则是让残量与 A 作用下的 Krylov 子空间正交。也就是说，GMRES 方法要求

$$(A\mathcal{K}_k(r_0))^\perp \ni r_k = r_0 - A\Delta x_k.$$

为了让读者更好地理解这两种方法的区别，请看下面的图示：

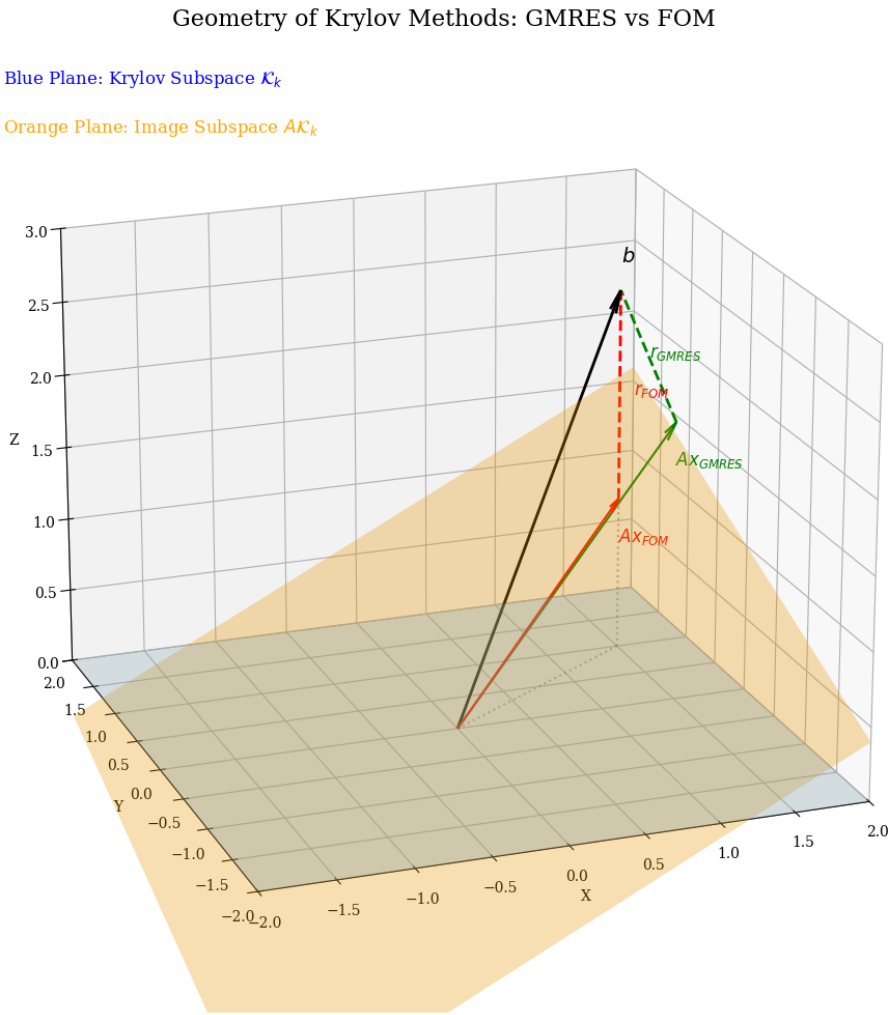

图 1: GMRES 方法与 FOM 方法的区别示意图

2.3 预处理

大多数迭代法的收敛速度与矩阵 A 的谱性质，尤其是条件数 $\kappa(A)$ 密切相关。对于比较病态的矩阵，我们往往需要通过预条件来改造系统。在实际计算中，Krylov 子空间方法往往需要预处理才能取得良好的收敛性。预处理的基本思想是：我们并不直接求解 $Ax = b$，而是求解一个与之等价但条件数更小的线性方程组 $M^{-1}Ax = M^{-1}b$，其中 M 是一个易于求解且近似于 A 的矩阵。这样处理的好处在于，$M^{-1}A$ 接近单位阵，不变子空间结构极其简单。而 Krylov 子空间本质上就是在逼近由特征向量张成的主不变子空间，因此 Krylov 子空间方法在预处理后的系统上往往能取得更好的收敛性。

常见的预处理方法有：

$$\begin{aligned}
&\text{左预处理：} && M^{-1}Ax = M^{-1}b, && \\
&\text{右预处理：} && AM^{-1}y = b, && x = M^{-1}y, \\
&\text{双侧预处理：} && M_1^{-1}AM_2^{-1}z = M_1^{-1}b, && x = M_2^{-1}z.
\end{aligned}$$

3 最速下降法

3.1 最速下降法

现在我们考虑一种特殊的线性系统：$Ax = b$，其中 A 是对称正定矩阵。我们知道，优化问题和求解线性方程组在某种程度上是等价的：例如在本例中，我们完全可以把 $Ax - b$ 看成是某个多元函数的梯度——这样我们只要求出使得原函数取得极值的点，就可以求出线性方程组的解了。

我们考虑 $\phi(x) = \frac{1}{2}x^{\mathrm{T}}Ax - b^{\mathrm{T}}x$，于是它的导数为 $\phi'(x) = Ax - b$，二阶导数为 $\phi''(x) = A \succ 0$. 因此 $\phi'(x)$ 的零点就是 $\phi(x)$ 的最小值点，而这个点的坐标就是线性方程组的解。因此，求解线性方程组 $Ax = b$ 等价于求解优化问题

$$\min_x \phi(x).$$

现在我们来考虑求解这个优化问题。我们不妨考虑迭代法：给定初值 x_0，我们希望通过迭代得到一个序列 $\{x_0, x_1, \ldots\}$——我们最朴素的愿望就是让这个序列越来越接近真实解，即 $\phi(x_0) > \phi(x_1) > \cdots$ 也就是说，我们要让函数序列 $(\phi(x_k))_{k\in\mathbb{N}}$ 严格单调下降。那么，我们该怎么让这个序列严格单调下降至最小呢？

想象一下，有一个盲人拿着一个登山杖，他的目标是找到山谷的最低点。他可以通过杖尖触地来感知地形的高低。每次他都可以选择一个方向前进，直到杖尖触地的高度不再下降为止。然后，他就可以在这个点停下来，认为自己找到了山谷的最低点。他每一步都试图找到一个下降最快的方向。找到方向以后，他还需确定一个步长——走得太快容易偏离最小方向，走得太慢又容易浪费时间。在确定方向和步长以后，他就可以前进了。这个过程就是最速下降法 (Steepest Descent Method,

SD) 的核心思想。这里的“下降最快的方向”，其实就是梯度的反方向，而步长则需要我们进一步确定。

于是，我们确定了 $x_k = x_{k-1} + \Delta x$，其中 $\Delta x = -\phi'(x) \cdot \alpha := r \cdot \alpha$，其中 $\alpha > 0$ 是每一步的步长。现在，我们唯一的任务就是确定这个步长，使得每一步的下降幅度最大。也就是说，我们要求解 $\min_\alpha \phi(x + r\alpha)$. 考虑

$$\begin{aligned}\phi(x + r\alpha) &= \frac{1}{2}(x + r\alpha)^{\mathrm{T}} A(x + r\alpha) - b^{\mathrm{T}}(x + r\alpha) \\ &= \frac{1}{2}x^{\mathrm{T}}Ax + r^{\mathrm{T}}Ax\alpha + \frac{1}{2}r^{\mathrm{T}}Ar\alpha^2 - b^{\mathrm{T}}x - b^{\mathrm{T}}r\alpha \\ &= \frac{1}{2}r^{\mathrm{T}}Ar\alpha^2 + (r^{\mathrm{T}}Ax - b^{\mathrm{T}}r)\alpha + \mathrm{Const}\end{aligned}$$

显然这是一个关于 α 的二次函数，因此我们很容易能够得到它的最小值点：

$$\alpha = \frac{r^{\mathrm{T}}(b - Ax)}{r^{\mathrm{T}}Ar} = \frac{r^{\mathrm{T}}r}{r^{\mathrm{T}}Ar} > 0.$$

于是，我们可以得到最朴素的最速下降法算法：

Algorithm 1: 利用最速下降法求解 $Ax = b$

Input: 对称正定矩阵 $A \in \mathbb{R}^{n\times n}$，向量 $b \in \mathbb{R}^n$，初值 $x_0 \in \mathbb{R}^n$，容忍度 $\varepsilon > 0$

Output: 线性方程组 $Ax = b$ 的近似解 x

```
x = x_0;
r = b − Ax ;                                   // 计算初始残差
while ‖r‖ > ε do
    α = (r^T r) / (r^T A r) ;                  // 计算步长
    x = x + αr ;                               // 更新解
    r = b − Ax ;                               // 更新残差
return x;
```

整个算法抽象出来其实就三个式子：

$$\begin{aligned}r_{k+1} &= b - Ax_k \\ \alpha_k &= \frac{r_k^{\mathrm{T}} r_k}{r_k^{\mathrm{T}} A r_k} \\ x_{k+1} &= x_k + \alpha_k r_k.\end{aligned}$$

现在，我们来分析这个算法的效率。这个算法涉及大量的向量内积运算和矩阵–向量乘法运算。每一步迭代中，计算残差 r_k 需要一次矩阵–向量乘法，计算步长 α_k 需要两次向量内积运算和一次矩阵–向量乘法，而更新解 x_k 则需要一次向量加法和一次数乘。因此，每一步迭代的计算复杂度主

要由矩阵–向量乘法决定——矩阵–向量乘法一般需要 $O(n^2)$ 的操作数，而每一步迭代我们都需要做两次这样的操作，这显然是很费时的。那么，我们能否进一步优化呢？我们不妨考虑

$$r_{k+1} = b - Ax_{k+1}$$
$$r_k = b - Ax_k$$

两式相减，有

$$r_{k+1} = r_k - A(x_{k+1} - x_k) = r_k - \boxed{Ar_k}\alpha_k.$$

细心的读者可以发现，被笔者框起来的 Ar_k 在计算步长 α_k 时被复用了。因此，在每一步迭代中，我们只需计算一次矩阵–向量乘法 Ar_k，然后把它存起来不断复用即可。

现在，我们来分析一下这个算法的收敛性。注意到在迭代过程中我们每一次都多乘了一个 A，因此直观上每一步的增量 $r_k\alpha_k$ 都会落在子空间 $\mathrm{span}\{r_0, Ar_0, \ldots, A^k r_0\}$ 内——这本质上是一种比较差的 Krylov 子空间方法，有时它的迭代步数会远超 n 步，而经典的 Krylov 子空间方法是能够保证在 n 步以内收敛的。

现在，我们来证明我们的直观猜想。我们用归纳法。显然 $\mathrm{span}\{r_0\} = \mathrm{span}\{r_0\}$. 假设对于某个 $k \geq 0$，我们已经有

$$\mathrm{span}\{r_0, Ar_0, \ldots, A^k r_0\} = \mathrm{span}\{r_0, r_1, \ldots, r_k\}.$$

接下来我们考虑 $k+1$ 的情况。由 $r_{k+1} = r_k - Ar_k\alpha_k$ 立即有 $r_{k+1} \in \mathrm{span}\{r_k, Ar_k\}$，而 $r_k \in \mathrm{span}\{r_0, Ar_0, \ldots, A^k r_0\}$. 因此

$$\mathrm{span}\{r_0, r_1, \ldots, r_k\} = \mathrm{span}\{r_0, Ar_0, \ldots, A^k r_0\}. \qquad \square$$

前面我们说最速下降法的迭代次数可能远超 n 步。下面我们就来解释一下为什么。我们知道，最速下降法每一步都沿着梯度的反方向前进，这实际上是一种贪心策略。虽然这种策略在每一步都能取得局部的最大下降，但它并不能保证整体的最优性。请看下图：

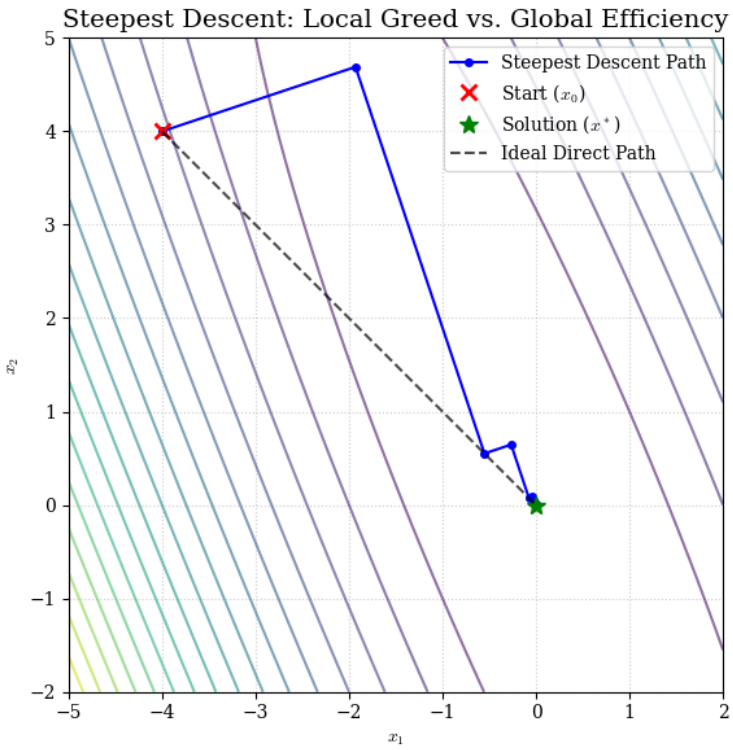

图 2: 最速下降法的迭代路径示意图

可以看到，最速下降法的迭代路径是一种锯齿形的路径。从直观上讲，这一点非常好理解。简单来说，正交性是当前这一步做到极致的必然代价。想象一下，你在山坡上，站在点 x_k，选择了负梯度方向 r_k 走到了这个方向上的最低点 x_{k+1}. 此时该点的切线必然与 r_k 平行，而新一步的方向 r_{k+1} 必须要与 r_k 垂直，否则你在 r_k 方向上还可以继续下降，这就与“走到该方向上的最低点”矛盾了。下面我们来严格地证明这一点。

我们在第 k 步迭代中求解是的这样一个子问题：

$$\min_{\alpha} \phi(x_k + r_k\alpha).$$

令 $g(\alpha) = \phi(x_k + r_k\alpha)$，这是一个关于标量 α 的函数，极值条件要求导数为零。利用链式法则，我们有

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\alpha} g(\alpha) &= \phi'(x_k + r_k\alpha)^{\mathrm{T}} \cdot r_k \\
&= \phi'(x_{k+1})^{\mathrm{T}} \cdot r_k \\
&= -r_{k+1}^{\mathrm{T}} r_k \quad \text{负梯度方向就是下降方向} \\
&= 0.
\end{aligned}$$

因此，我们有 $r_{k+1}^{\mathrm{T}} r_k = 0$，即梯度在相邻两步之间是正交的——在上面的证明中我们并没有使用 ϕ 的具体形式，因此正交性是线优化的必然结果。这就解释了为什么最速下降法的迭代路径会呈现

出锯齿形。在狭长的山谷中，这种正交性迫使我们只能以直角转弯前进，就像在走楼梯。再看图中的黑色虚线。这是起点 x_0 到终点 $x_\star$ 的最优路径。如果算法有全局视野，它应该沿着这条黑色虚线走。最速下降法的悲剧在于：局部下降最快的方向，通常并不指向圆心。矩阵越病态（等高线越扁），梯度方向偏离圆心的角度就越大，锯齿效应就越严重。正交性是贪婪的代价。因为我们在当前方向上榨干了所有的下降潜力，导致新生成的残差向量必须与当前方向正交。这种强制的正交性，使得我们无法保留惯性，只能在山谷中走直角弯。

一句话总结，最速下降法比常见的 Krylov 子空间方法慢的本质原因在于，它们虽然搜索空间相同，但 GMRES 和 FOM 方法保证了全局的最优性，而最速下降法只保证了局部的最优性。

3.2 Krylov 子空间方法的几何本质：范数的较量

在数值线性代数中，我们经常听到“最小化”这个词。但这里的关键问题是：我们在最小化什么？在哪个空间里最小化？不同的迭代法，本质上是在不同的度量空间（范数）里寻找最优解。

对于一个线性方程组 $Ax = b$，我们可以定义误差向量 $e_k = x_\star - x_k$，其中 $x_\star$ 是方程组的精确解，这是我们真正关心的量，但通常不好算；定义残差向量 $r_k = b - Ax_k$，这是可以计算的量。显然我们有 $r_k = Ae_k$.

GMRES 的优化目标是最小化残差的 2-范数：

$$x_k = \arg\min_{x \in \mathcal{K}_k(b)} \|b - Ax_k\|_2 = \arg\min_{x \in \mathcal{K}_k(b)} \|r_k\|_2.$$

它的几何意义是：在 A 作用下的 Krylov 子空间中寻找与 b 最接近的点。这是一种比较务实的策略：既然我们只能看到残差，那就让它越小越好。

最速下降法和下节课要讲的共轭梯度法处理的都是对称正定矩阵。在这种情况下，我们定义一种新范数 $\|x\|_A = \sqrt{x^{\mathrm{T}} A x}$. 这两种方法本质上是在最小化误差的 A-范数：

$$x_k = \arg\min_{x_k \in \mathcal{K}_k(b)} \|x_\star - x_k\|_A.$$

为什么这么说呢？回想我们定义的二次函数 $\phi(x) = \frac{1}{2} x^{\mathrm{T}} A x - b^{\mathrm{T}} x$，我们考虑

$$\begin{aligned}
\phi(x_k) - \phi(x_\star) &= \frac{1}{2} x_k^{\mathrm{T}} A x_k - b^{\mathrm{T}} x_k - \left(\frac{1}{2} x_\star^{\mathrm{T}} A x_\star - b^{\mathrm{T}} x_\star \right) \\
&= \frac{1}{2} x_k^{\mathrm{T}} A x_k - x_k^{\mathrm{T}} A x_\star - \left(\frac{1}{2} x_\star^{\mathrm{T}} A x_\star - x_\star^{\mathrm{T}} A x_\star \right) \quad \text{因为 } Ax_\star = b \\
&= \frac{1}{2} (x_k - x_\star)^{\mathrm{T}} A (x_k - x_\star) \\
&= \frac{1}{2} \|x_k - x_\star\|_A^2.
\end{aligned}$$

因此，我们每一步操作都是在 Krylov 子空间 $\mathcal{K}_k(b)$ 上寻找使得 $\|\phi(x) - \phi(x_\star)\|_A$ 最小的点。