GMRES - Overview and Algorithms

1 Basic

Algorithm 1: GMRES

Input: A, b, maxiter, tol
Output: x, $\|r\|$

x_0 = Initial Guess
$r_0 = b - Ax_0$
m = maxiter
`// First Krylov vector` q_1 `with` $\|q_1\| = 1$
$q_1 = r_0/\|r_0\|$
for *j=1..m* **do**
  `// (1) Generate Krylov vector` q_{j+1}
  $v = Aq_j$
  for *i=1..j* **do**
    $h_{i,j} = q_i^T v$
    $v = v - h_{i,j} q_i$
  $h_{j+1,j} = \|v\|$
  $q_{j+1} = v/h_{j+1,j}$
  Add q_{j+1} as column to $Q_j \to Q_{j+1}$
  `// (2) Find search vector` y_j`, where` $x = x_0 + Q_j y_j$
  min $\|\beta e_1^{j+1} - \tilde{H}_j y\|$ for y_j, where $\tilde{H}_j = \{h_{i,j}\}_{1 \le j+1, 1 \le j}$, $\beta = \|r_0\|$, and $e_1^{j+1} = [1, 0, .., 0]^T \in \mathbb{R}^{j+1}$
  `// (3) Check residual`
  $x = x_0 + Q_j y_j$
  $r = b - Ax$
  if $||r|| < tol$ **then**
    break

References: [1]

2 Solving minimization problem with Givens rotation

One part of the GMRES is to find the vector y_j which minimizes

$$\min \|\tilde{H}_j y_j - \beta e_1\| \tag{1}$$

where $\tilde{H}_j$ is an $(j+1) \times j$ upper Hessenberg Matrix. We can use Givens rotation to transfer (1) into an upper triangular system of equations of order j. A Givens rotation matrix J_i with order $j+1$ is an identity matrix where only four elements are replaced as follows

$$\begin{pmatrix} 1 & & & 0 \\ & c_i & s_i & \\ & -s_i & c_i & \\ 0 & & & 1 \end{pmatrix}$$

such that $\begin{pmatrix} c_i & s_i \\ -s_i & c_i \end{pmatrix} \begin{pmatrix} h_{i,i} \\ h_{i+1,i} \end{pmatrix} = \begin{pmatrix} * \\ 0 \end{pmatrix}$. By multiplying the product of j Givens rotations $\Omega_j = J_j J_{i-1}..J_1$ from the left-hand side to $\tilde{H}_j$ we obtain a triangular system

$$\tilde{H}_j = \begin{pmatrix} * & * & * & * \\ * & * & * & * \\ & * & * & * \\ & & * & * \\ & & & * \end{pmatrix}, \qquad \Omega_j \tilde{H}_j = \begin{pmatrix} + & + & + & + \\ 0 & + & + & + \\ & 0 & + & + \\ & & 0 & + \\ & & & 0 \end{pmatrix} = \begin{pmatrix} R_j \\ 0 \end{pmatrix}, \quad (R_j \in \mathbb{R}^{j \times j}).$$

Multiplying the Givens product Ω_j to βe_1 gives

$$\Omega_j \beta e_1 = \begin{pmatrix} g_j \\ \lambda_j \end{pmatrix}, \quad (g_j \in \mathbb{R}^j, \lambda_j \in \mathbb{R}^1).$$

Thus multiplying Ω_j to the minimization problem (1), we find

$$\min \|\Omega_j \tilde{H}_j y_j - \Omega_j \beta e_1\| = \min \|R_j y_j - g_j\|. \tag{2}$$

In the code, we now store R_j instead of $\tilde{H}_j$. At each GMRES iteration, we extend $\tilde{H}_j \to \tilde{H}_{j+1}$ by a new row and a new column, say $\tilde{h} \in \mathbb{R}^{j+2}$. Since we store R_j instead of $\tilde{H}_j$, we only need to apply the Givens rotations Ω_j to $\tilde{h}$ and add the new column to $R_j \to R_{j+1}$. We do not store the rotation matrices J_j explicitly but keep c_j and s_j from each iteration.

2.1 Residual Norm

From (2) we get y_j, from which we can update $x = x_0 + Q_j y_j$ and calculate the residual norm $\|b - Ax\|$ to assess whether our algorithm has converged. In this case, we would solve the minimization problem (2) in each GMRES iteration. However, there is a better way to get the residual norm, without explicitly calculating y_j and x at each iteration, see also [2] and [1] §6.5.3.

Substituting $\Omega_j \tilde{H}_j = \begin{pmatrix} R_j \\ 0 \end{pmatrix}$ and $\Omega_j \beta e_1 = \begin{pmatrix} g_j \\ \lambda_j \end{pmatrix}$ in (2) gives

$$\|\Omega_j \tilde{H}_j y_j - \Omega_j \beta e_1\| = \|\begin{pmatrix} R_j \\ 0 \end{pmatrix} y_j - \begin{pmatrix} g_j \\ \lambda_j \end{pmatrix}\| = \|R_j y_j - g_j\| + \|\lambda_j\|.$$

Since $\|R_j y_j - g_j\|$ vanishes by construction from (2), the residual norm is thus the absolute value of the last component of $\Omega_j \beta e_1$, i.e. λ_j. Therefore, the residual norm is available at no extra cost at each step and we only need to explicitly compute y_j and x after the GMRES algorithm has converged. The GMRES algorithm with Givens rotation and the non-explicit residual method is shown in Algorithm 2.

References

[1] Y. Saad. *Iterative Methods for Sparse Linear Systems.* Second. Society for Industrial and Applied Mathematics, 2003.

[2] Y. Saad and M. H. Schultz. "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems". In: *SIAM J. Sci. Stat. Comput* 7 (1986).

Algorithm 2: GMRES - Least-squares problem solved with Givens rotation

Input: A, b, maxiter, tol
Output: x

x_0 = Initial Guess
$r_0 = b - Ax_0$
m = maxiter
// First Krylov vector q_1 with $\|q_1\| = 1$
$q_1 = r_0/\|r_0\|$
$g_1 = \|r_0\|$
for *j=1..m* **do**
  // (1) Generate Krylov vector q_{j+1}
  $v = Aq_j$
  for *i=1..j* **do**
    $\tilde{h}_i = q_i^T v$
    $v = v - \tilde{h}_i q_i$
  $\tilde{h}_{j+1} = \|v\|$
  $q_{j+1} = v/\tilde{h}_{j+1}$
  // (2) Apply Givens rotation to $\tilde{h}$ and βe_1
  $\tilde{r}_j$, c_j, s_j = ApplyGivRot($\tilde{h}$, $\{c_j\}_{1<j}$, $\{s_j\}_{1<j}$)
  $\lambda_j = -s_j g_j$
  $g_j = c_j g_j$
  // (3) Check residual
  if $||\lambda_j|| < tol$ **then**
    break
  $g_{j+1} = \lambda_j$
min $\|g - R_j y\|$
$x = x_0 + Q_j y_j$

Algorithm 3: GivRot - Calculate Givens matrix components, $\begin{pmatrix} c_i & s_i \\ -s_i & c_i \end{pmatrix} \begin{pmatrix} f \\ g \end{pmatrix} = \begin{pmatrix} * \\ 0 \end{pmatrix}$

Input: f, g
Output: c, s

$c = f/\sqrt{f^2 + g^2}$
$s = g/\sqrt{f^2 + g^2}$

Algorithm 4: ApplyGivRot - Apply Givens rotations product to a vector $\mathbf{v}$, i.e. $\mathbf{v} \to \Omega_j \mathbf{v} = (J_j J_{i-1}..J_1)\mathbf{v}$

Input: $\mathbf{v} \in \mathbb{R}^{k+2}$, Givens components $\mathbf{c} \in \mathbb{R}^k$ and $\mathbf{s} \in \mathbb{R}^k$
Output: Updated $\mathbf{v}$, k-th Givens matrix components c_k, s_k

// Apply for i-th column
for *i=1..k-1* **do**
  $\begin{pmatrix} v_i \\ v_{i+1} \end{pmatrix} = \begin{pmatrix} c_i & s_i \\ -s_i & c_i \end{pmatrix} \begin{pmatrix} v_i \\ v_{i+1} \end{pmatrix}$
// Update the next cos/sin values for rotation
c_k, s_k = GivRot(v_k, v_{k+1})
// Eliminate v_{k+1}
$v_k = c_k v_k + s_k v_{k+1}$
$v_{k+1} = 0$