QR with column pivoting

Abstract

Algorithms for computing QR factorizations with column pivoting

1. QR with column pivoting

Algorithm 1 Businger and Golub QR with column pivoting

procedure BAG-QRPivoting(A)
  Compute and store column norms of A in `columnnorms`
  Declare permutation vector, `perm=1:n`
  for `k=1:n-1` **do**
    Determine maximum column norm index, `jmax`.
    Swap $\mathbf{a}_k$ with $\mathbf{a}_{jmax}$ ▷ Swaping
    Swap `columnnorms(k)` with `columnnorms(jmax)`
    Swap `perm(k)` with `perm(jmax)`
    Compute $\mathbf{v}, \beta$ for $\mathbf{a}(k:m,k)$ ▷ Householder reflection
    Compute $\boldsymbol{H}_k = \boldsymbol{I}_{m-k+1} - \beta \mathbf{v}\mathbf{v}^T$
    Update $\boldsymbol{A}(k:m, k+1:n) = \boldsymbol{H}_j \boldsymbol{A}(k:m, k+1:n)$
    `columnnorms(k+1:n) = columnnorms(k+1:n) -` $\boldsymbol{A}(k, k+1:n)^2$
  end for
  $\boldsymbol{R}$ =`triu`($\boldsymbol{A}$)
  $\boldsymbol{Q} = \boldsymbol{H}_{n-1} \ldots \boldsymbol{H}_1 \boldsymbol{A}$
  return $\boldsymbol{Q}$, $\boldsymbol{R}$, `perm`
end procedure

Algorithm 2 QR with column pivoting for $\boldsymbol{D} \in \mathbb{R}^{m \times n}$ where $m < n$

1: **procedure** MGS-QRPivoting($\boldsymbol{D}$)
2: Compute and store column norms of $\boldsymbol{D}$ in `colnorms`
3: Declare permutation vector, `perm=1:n`
4: **for** `k=1:m` **do**
5: Store `max(colnorms(k:n))` index as `pmax`.
6: Swap $\mathbf{d}_k$ with $\mathbf{d}_{jmax}$ ▷ Swaping
7: Swap `colnorms(k)` with `colnorms(jmax)`
8: Swap `perm(k)` with `perm(jmax)`
9: **if** k is not equal to 1 **then**
10: **for** `i=1:k-1` **do**
11: $\mathbf{d}_k = \mathbf{d}_k - \mathbf{q}_i^T \mathbf{d}_k \mathbf{q}_i$ ▷ Reorthogonalization
12: **end for**
13: **end if**
14: $\mathbf{q}_k = \mathbf{d}_k / \|\mathbf{d}_k\|_2$ ▷ Normalization
15: **if** k is not equal to n **then** ▷ Orthogonalization
16: **for** `j=k+1:n` **do**
17: $\mathbf{d}_j = \mathbf{d}_j - \mathbf{q}_k^T \mathbf{d}_j \mathbf{q}_k$
18: Update `colnorms(j)`
19: **end for**
20: **end if**
21: **end for**
22: **return** `perm(1:m)`
23: **end procedure**