MAP55672 (2024-25) — Case studies 2

2.1 Arnoldi Iteration

Given matrix A and initial vector u, construct the orthogonal basis matrix Q of the Krylov subspace and generate the corresponding upper Hessenberg matrix H.

Implementation:

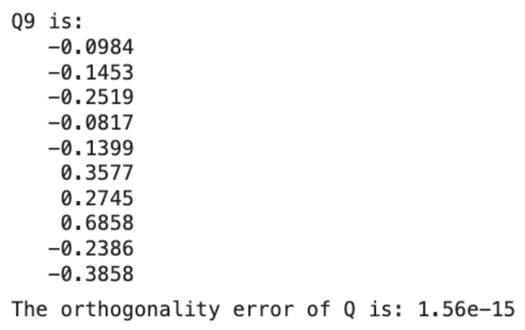
1. **Initialization**: Normalize the initial vector u, and use it as the first column of Q.
2. **Iterative Construction**: For $j = 1$ to m:
   - Compute $w = AQ(:, j)$
   - Orthogonalize against the existing basis vectors $Q(:, 1), \ldots, Q(:, j)$:
     $h_{i,j} = Q(:, i)^\top w, \quad w = w - h_{i,j} Q(:, i)$
   - Set $h_{j+1,j} = \|w\|$. If it is very small, terminate the iteration early; otherwise, normalize $w / h_{j+1,j}$ as $Q(:, j+1)$.

   ```
   v = A * q_j
   for i = 1 to j
       h_ij = q_i' * v
       v = v - h_ij * q_i
   end
   h_{j+1,j} = norm(v)
   q_{j+1} = v / h_{j+1,j}
   ```

3. **Output Results**:
   - $Q \in \mathbb{R}^{n \times (m+1)}$ is the orthogonal basis matrix;
   - $H \in \mathbb{R}^{(m+1) \times m}$ is the upper Hessenberg matrix, recording the projection coefficients at each step.

Testing and Verification:

- Test on a given 10×10 matrix, with $m = 9$;
- Extract the 10th column of Q, and verify if the first 9 columns are orthogonal to confirm the correctness of the implementation.
- Output：



```
Q9 is:
   -0.0984
   -0.1453
   -0.2519
   -0.0817
   -0.1399
    0.3577
    0.2745
    0.6858
   -0.2386
   -0.3858
The orthogonality error of Q is: 1.56e-15
```

2.2 Serial implementation of GMRES.

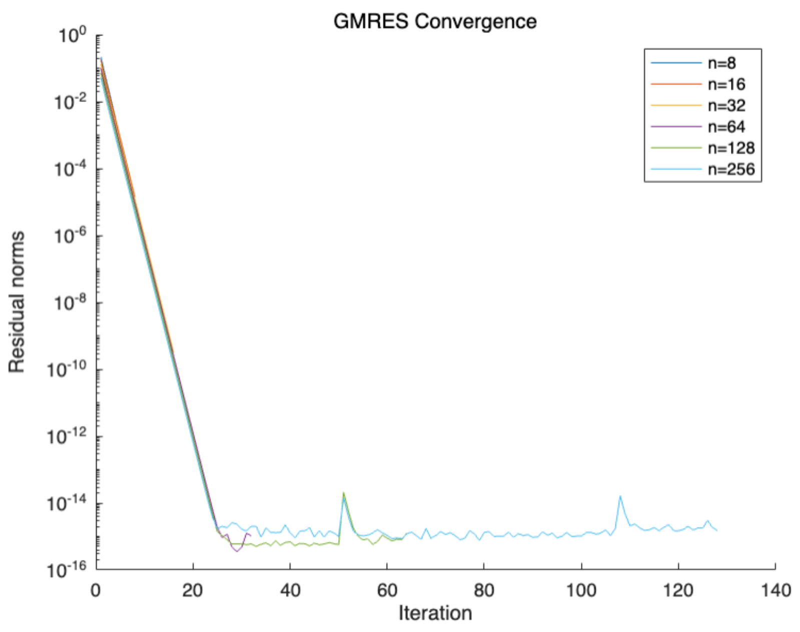
Implementation and Testing Instructions

Function gmres(A, b, m)

- **Input:** Matrix A, vector b, and maximum number of iterations m.
- **Output:** Final solution x and the residual history for each iteration.
- **Method:**
  1. Start with initial guess $x_0 = 0$, compute residual $r_0 = b - Ax_0$
  2. Use Arnoldi to generate orthonormal basis Q and Hessenberg matrix H
  3. Solve least squares problem $\min \|\beta e_1 - Hy\|$ at each iteration
  4. Construct approximate solution $x = Qy$ and record residual norm
  5. Stop early if residual norm is below threshold

Matrix and vector construction:

- **build_matrix(n):** Constructs a tridiagonal matrix with -4 on the diagonal and 1 on the upper and lower diagonals.
- **build_b(n):** Returns a vector b as the right-hand side.



From the graph, we can observe that the residuals have dropped to a very small level (around 10^{-15}).

As n increases from 8 to 256, the convergence curve drops below 10^{-10} around the 20th to 30th iteration, and then fluctuates within a very small range. This shows that the GMRES convergence rate does not strongly depend on n in this case.

As n increases, some numerical fluctuations may appear in later iterations, but overall, the convergence is still fast..

2.3 Parallel implementation of GMRES.

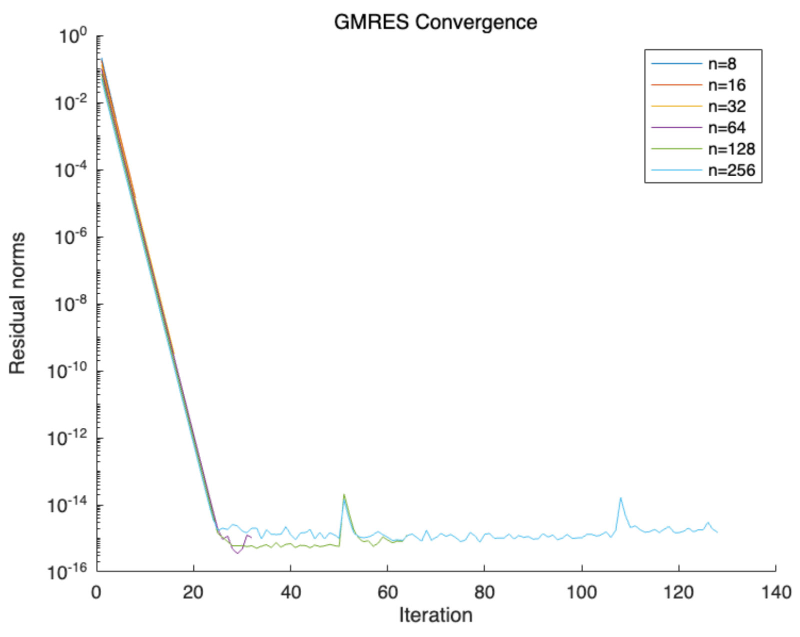
- Based on serial GMRES in 2.2, parallelize the most time-consuming matrix-vector multiplication part using `parfor`.

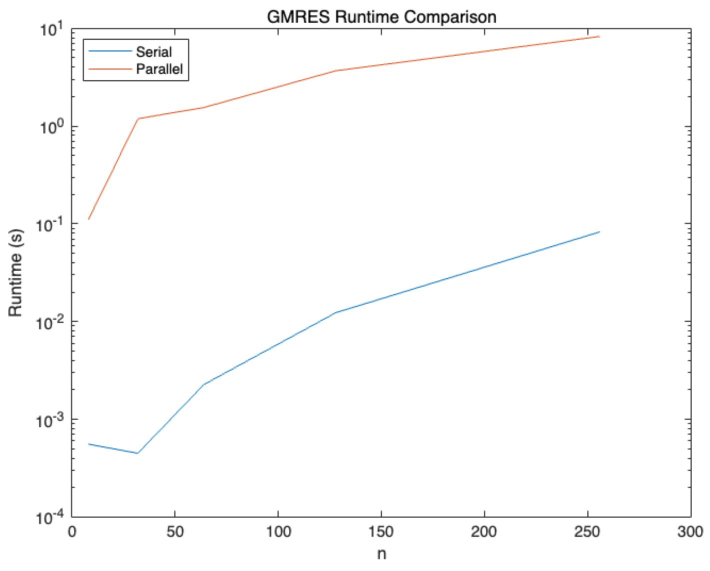
  command:

  ```
  >> parpool;
  Starting parallel pool (parpool) using the 'Processes' profile ...
  Connected to parallel pool with 8 workers.
  ```

- The other steps (orthogonalization, least squares solving, residual recording) remain serial.
- In the j-th step of the Arnoldi iteration, replace `w = A * Q(:,j)` with:

  ```
  w = zeros(n, 1);
  parfor i = 1:n
      w(i) = A(i,:) * Q(:,j);
  end
  ```





From the graph, parallel implementation is actually slower than serial in the current environment.

The main reason is the problem size is small, and it cannot offset the overhead of parallel scheduling. To fully utilize parallelism, testing should be done on a larger scale or on multi-core/multi-node platforms.

Correctness

- Use the same test cases as in 2.2, with problem sizes n = 8, 32, 64, 128, 256
- Compare the two residual history curves. The parallel and serial versions are almost identical in both the residual convergence curve and the final solution, indicating that the parallel implementation is correct.

Stopping Criterion

- Stop when the residual $||r_k||$ (or normalized residual) is below a certain tolerance (such as $1e-8$ or $1e-14$); or when the iteration count reaches the maximum m.