Solving System of Linear Equation with Iterative method

Pengfei Cui

System of Linear Equation

- Ax=b
- A is an n*n matrix
- b is an n dimensional vector

Iterative method

- $Ax=b \rightarrow (M-N)x = b \rightarrow Mx = Nx+b$
- $x = M^{-1}Nx + M^{-1}b$
- $Dx = (L + U)x + b$
- $x = D^{-1} (L + U)x + D^{-1}b$

Jacobi & Gauss-Seidel Method

- $x_i^{k+1} = \frac{1}{a_{ii}}\left(b_{ii} - \sum_{i \neq j} a_{ij} x_i^k\right)$
- $x_i^{k+1} = \frac{1}{a_{ii}}\left(b_{ii} - \sum_{j<i} a_{ij} x_i^{k+1} - \sum_{j>i} a_{ij} x_i^k\right)$

Diagonally Dominant Matrix

- $|a_{ii}| \geq \sum_{\{j \neq i\}} |a_{ij}|$

MPI Command

- MPI_Send
- MPI_Receive
- MPI_Bcast
- MPI_Allgather
- MPI_Allreduce

Parallel

- Divide the matrix by row

Implement

- Input
- Calculate
- Output

Input

- Processor 0 read A and b
- Processor 0 broad A and b to other processor
- Each processor only need part of A and b

```
if (rank == 0){
    //send
    for (i=0;i<m;i++){
        for (j=0;j<n;j++){
            a[i][j] = ra[i][j];
        }
        b[i]=rb[i];
    }

    for (i=m;i<n;i++){
        MPI_Send(&ra[i][0], n, MPI_FLOAT, i/m, 0, MPI_COMM_WORLD);
    }

    for (i=1;i<size;i++){
        MPI_Send(&rb[i*m], m, MPI_FLOAT, i, 0, MPI_COMM_WORLD);
    }
    free(ra);
    free(rb);
} else {
    //receive
    printf("rank=%d begin receive \n",rank);
    for (i=rank*m;i<(rank+1)*m && i<n;i++){
        MPI_Recv(&a[i][0], n, MPI_FLOAT, 0, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
    }
    MPI_Recv(&b[rank*m], m, MPI_FLOAT, 0, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
    printf("rank=%d finish receive \n",rank);
}
```

Calculate

- While (1) {
  - Calculate x;
  - get d;
  - If (d<e) break;
- }

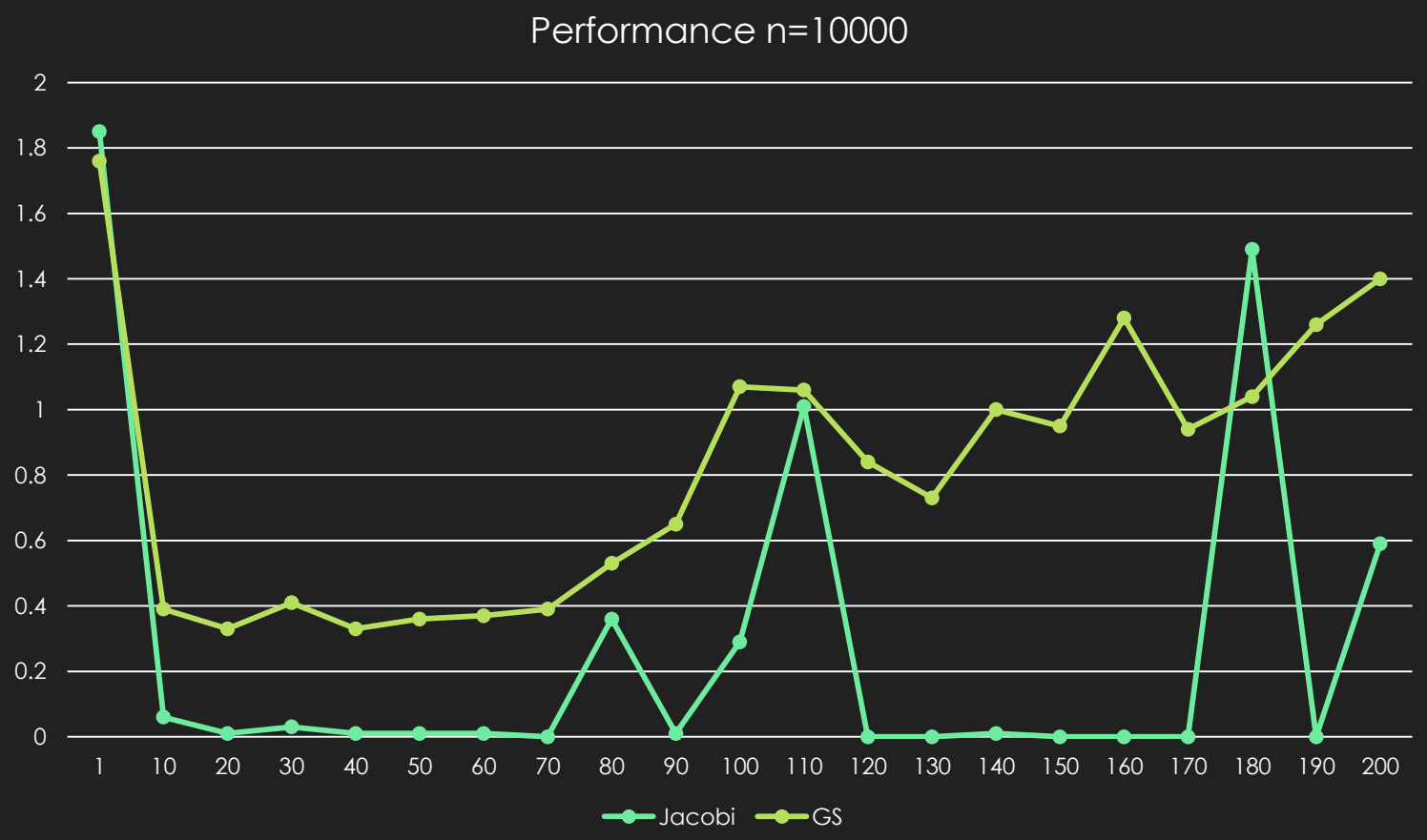
Calculate

- Jacobi
  - For (i=m*rank;i<(m+1)*rank;i++){
    - Calculate x[i]
  - }
  - Gather x
- Gauss-Seidel
  - For (i=0;i<n;i++) {
    - if (i/m == rank){
      - Caclulate x[i]
      - Send x[i]
    - } else {
      - Receive [xi];
    - }
  - }

Output

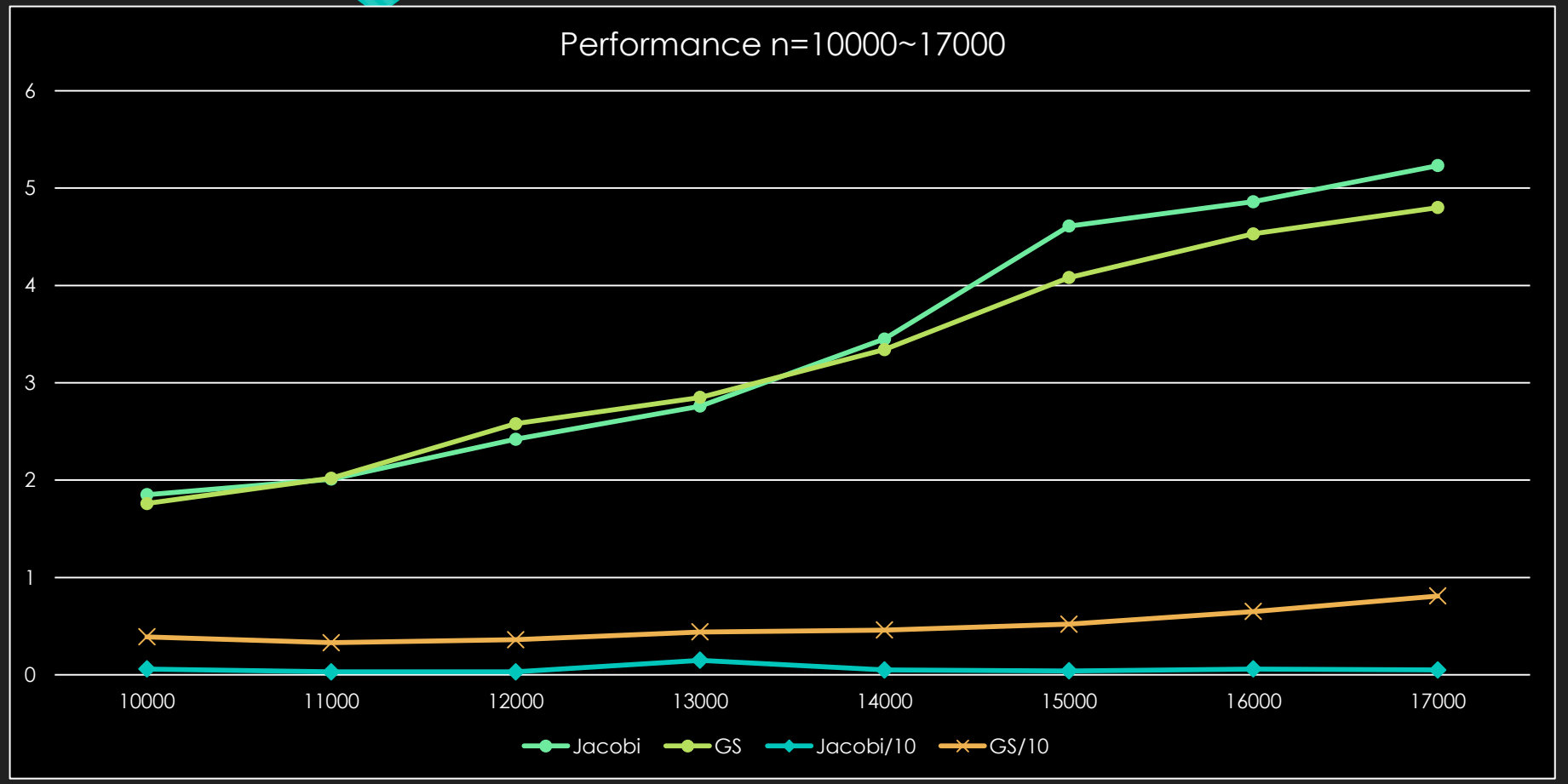
- Processor 0 output x

Performance

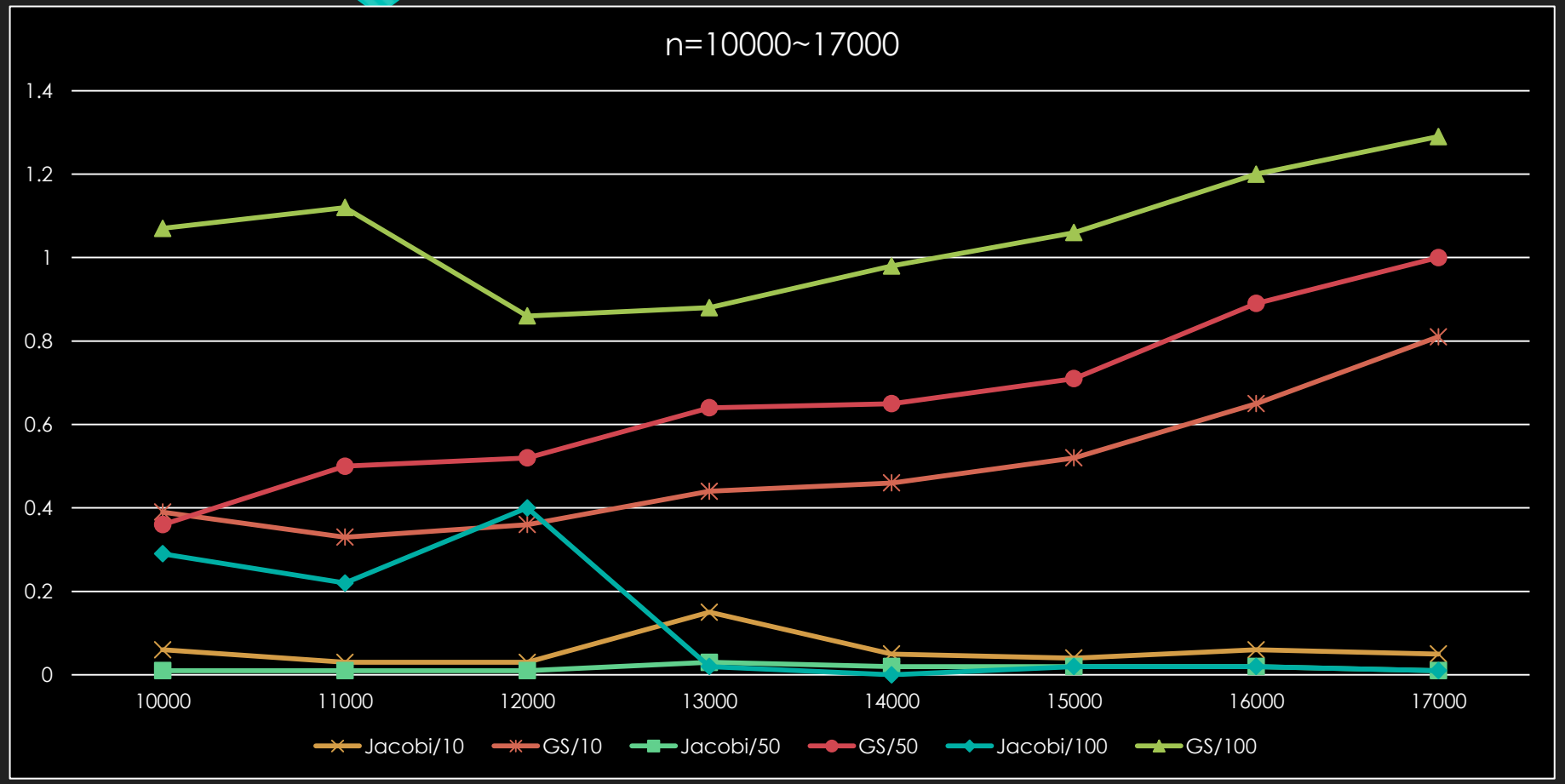


Performance

Performance



Performance



Further work

- Input take too much time
- The performance of Gauss-Seidel is not so good

Thank you