Matlab and Linear Systems

Raj K. Markanda
Department of Mathematics and Natural Sciences
Northern State University
Aberdeen, SD 57401
Tel: (605)626-2456
e-mail: Markandr@wolf.northern.edu

There are several direct, elimination, and iterative methods for solving systems of n equations in n variables. We will use MATLAB students version 4 to compare these methods. For large n the best methods for solving Ax = b are Gaussian elimination and Gauss-Jordan elimination methods each of which requires approximately $\frac{n^3}{3}$ multiplications and $\frac{n^3}{3}$ additions. The method of multiplication by A^{-1} is much worse than these and Cramer's rule is the worst of these 3 methods. For sparse matrices, Jacobi and Gauss-Seidel iteration methods are very useful because the zeros simplify the iteration equation, thereby reducing the amount of calculations.

Direct and elimination methods are commonly studied in a first year linear algebra course. The iterative methods generally appear at the end of most texts. We will illustrate Jacobi and Gauss-Seidel methods with the help of the following example:

Solve:

$$\begin{aligned} 20x_1 - x_2 + x_3 &= 20 \\ 2x_1 + 10x_2 - x_3 &= 11 \\ x_1 + x_2 - 20x_3 &= -18 \end{aligned}$$

We can rewrite these equations as:

$$y_1 = \frac{x_2 \quad x_3 \quad 20}{20}$$

$$y_2 = \frac{2x_1 \quad x_3 \quad 11}{10}$$

$$y_3 = \frac{x_1 \quad x_2 \quad 18}{20}$$

Jacobi Method:

We take $x^{(0)} = (0, 0, 0)$ as an initial approximation to the solution, and use four iterations to get:

$$x^{(1)} = (1, 1.1, 0.9)$$

$$x^{(2)} = (0.995, 0.97, 1.005)$$

$$x^{(3)} = (0.99825, 1.0015, 0.99825)$$

$$x^{(4)} = (1.0001625, 1.000175, 0.9999875)$$

Gauss-Seidel Method:

Take $x^{(0)} = (0, 0, 0)$

We calculate $x_1 = y_1$ using $x_2 = 0$, $x_3 = 0$. Then use this new x_1 along with $x_3 = 0$ to compute $x_2 = y_2$. Finally, in the third equation, use the new values for x_1 and x_2 to compute x_3 . Thus, we get:

$$x^{(1)} = (1, 0.9, 0.995)$$

$$x^{(2)} = (0.99525, 1.00045, 0.999785)$$

$$x^{(3)} = (1.00033, 0.9999719, 1.0000002)$$

This shows that this sequence converges faster than the sequence of Jacobi method. There are examples where the Jacobi method is faster than the Gauss-Seidel method. In general, both methods work when the coefficient matrix A is strictly diagonally dominant. See [2, page 292].

We use sparse matrices and random matrices of different sizes to compare these methods. We first solve:

$$Ax = b$$

with a symmetric, triagonal and sparse matrix A given by:

A(i, i) = 3 for 1 *i* *n*

A(i, i-1) = -1 for 2 *i* *n*

A(i-1, i) = -1 for 2 *i* *n*

Note that for n = 32, A has only 94 non-zero entries out of 1024 entries.

Let b be defined by b(i, 1) = 0.02i and the initial solution x defined by x(i, 1) =0 for 1 i n.

Take:

```
D = diag(diag(A))
P = D - A
L = tril(A)
U = triu(A) - D
```

Matlab procedure for Jacobi method:

```
for k = 1 : 10,  y = inv(D)(Px + b),  x = y; pause, end.
```

Matlab procedure for Gauss-Seidel method:

```
for k = 1 : 10,  y = inv(L)(-Ux + b), x = y; pause, end.
```

Number of flops for the matrix

Method	A(6,6)	A(12,12)	A(24,24)	A(32,32)
x=A\b	237	876	4079	15576
x=inv(A)*b	374	1472	6515	50911
x=rref([A b])	995	4316	22448	67048

Jacobi method	3520 (10 iter)	13540 (10iter)	37812 (6 iter)	529910 (10 iter)
G-S method	2884 (7 iter)	8090 (5 iter)	15382 (2 iter)	317946 (6 iter)

We now use A = random (n) + nI and repeat the above calculations.

<u>Number of flops</u>

Method	A(6,6)	A(12,12)	A(24,24)	A(36,36)
A\b	377	2089	12953	28723
inv(A)*b	654	4335	31125	71940
rref([A b])	2573	10856	50636	78756
Jacobi method	1760 (5 iter)	8124 (6 iter)	31812 (6 iter)	317946 (6 iter)
G-S method	1636 (3 iter)	8979 (3 iter)	56409 (3 iter)	219888 (3 iter)

Anton's linear algebra text has a chapter on numerical method of linear algebra where one can find discussion of computational time for various methods.

References

1. H. Anton, Elementary Linear Algebra, John Wiley, 1991.

2. David R. Hill, Experiments in Computational Matrix Algebra, Random House, 1988.