Solve a System of Equations in R (3 Examples)

Authored by

Mohammed looti

November 2, 2025

RECOMMENDED CITATION

Mohammed looti (2025). *Solve a System of Equations in R (3 Examples)*. PSYCHOLOGICAL STATISTICS. Retrieved from https://statistics.arabpsychology.com/?p=8384

The Power of R for System of Equations

Solving a system of linear equations is a fundamental task in mathematics, engineering, and data science. When dealing with multiple variables and complex systems, computational tools become essential. Within the statistical programming environment of R, we can utilize the highly efficient built-in function, **solve()**, designed specifically for matrix algebra operations.

This function leverages principles of linear algebra to quickly determine the unknown variables. The following guide provides a detailed breakdown of how to structure your equations as matrices and apply the **solve()** function across systems of increasing complexity, from two variables up to four.

Mathematical Foundation: Representing Linear Systems as Matrices

Before executing the solution in R, it is critical to understand the standard mathematical representation of a linear system. Any set of linear equations can be expressed in the form **AX = B**, where:

A is the coefficient matrix (containing the coefficients of the variables).
X is the vector of unknown variables (the solution we are seeking).
B is the constant vector (containing the values on the right-hand side of the equations).

The **solve()** function in R requires us to explicitly define the **A** matrix and the **B** vector as R objects. The solution **X** is then calculated by multiplying the inverse of **A** by **B** (i.e., $X = A^{-1}B$).

Example 1: Solving a System of Equations with Two Variables

We begin with a simple system involving two variables, x and y. This process demonstrates the core steps required for setting up and solving any linear system in R.

Consider the following system of equations:

5x + 4y = 35

2x + 6y = 36

To prepare this for the **solve()** function, we must extract the coefficients into the **A** matrix and the constants into the **B** vector. Notice how the coefficients are entered column-wise into the **matrix()** function in R.

```
#define left-hand side of equations
left_matrix <- matrix(c(5, 2, 4, 6), nrow=2)
```

```
left_matrix

5 4
2 6

#define right-hand side of equations
right_matrix <- matrix(c(35, 36), nrow=2)

right_matrix

35
36

#solve for x and y
solve(left_matrix, right_matrix)

3
5
```

The resulting output provides the values for the variables in the order they were defined in the coefficient matrix (A). This tells us that the value for x is **3** and the value for y is **5**. We can verify this solution by substituting these values back into the original equations.

Example 2: Expanding to a System of Equations with Three Variables

The methodology remains consistent when scaling up to higher dimensions. For a system involving three variables (x, y, and z), we will construct a $3 times 3$ coefficient matrix (A) and a $3 times 1$ constant vector (B).

Suppose we have the following system:

4x + 2y + 1z = 34

3x + 5y - 2z = 41

2x + 2y + 4z = 30

We define the coefficient matrix (A) by collecting the coefficients for x (column 1), y (column 2), and z (column 3). Note that coefficients of 1 or -1 must be explicitly included in the matrix definition within R.

```
#define left-hand side of equations
left_matrix <- matrix(c(4, 3, 2, 2, 5, 2, 1, -2, 4), nrow=3)
```

```
left_matrix

4 2 1
3 5 -2
2 2 4

#define right-hand side of equations
right_matrix <- matrix(c(34, 41, 30), nrow=3)

right_matrix

34
41
30

#solve for x, y, and z
solve(left_matrix, right_matrix)

5
6
2
```

The computation yields the solution vector. This tells us that the value for x is **5**, the value for y is **6**, and the value for z is **2**. This example clearly illustrates the power of **solve()** in handling multivariate linear algebra problems efficiently.

Example 3: Solving a System of Equations with Four Variables

For systems involving four or more variables, manual calculation becomes highly impractical. This is where computational tools like R truly shine. In this example, we solve for w, x, y, and z, requiring a $4 times 4$ coefficient matrix.

Suppose we have the following complex system of equations:

$$6w + 2x + 2y + 1z = 37$$

$$2w + 1x + 1y + 0z = 14$$

$$3w + 2x + 2y + 4z = 28$$

$$2w + 0x + 5y + 5z = 28$$

When constructing the coefficient matrix (A), ensure that coefficients of zero (as seen for z in the

second equation and x in the fourth equation) are included as the number 0 in the vector passed to the **matrix()** function. This ensures the correct structure for the $4 times 4$ matrix.

```
#define left-hand side of equations
left_matrix <- matrix(c(6, 2, 3, 2, 2, 1, 2, 0, 2, 1, 2, 5, 1, 0, 4, 5), nrow=4)

left_matrix

6 2 2 1
2 1 1 0
3 2 2 4
2 0 5 5

#define right-hand side of equations
right_matrix <- matrix(c(37, 14, 28, 28), nrow=4)

right_matrix

37
14
28
28

#solve for w, x, y and z
solve(left_matrix, right_matrix)

4
3
3
1
```

The resulting matrix provides the solution vector for the four variables. This tells us that the value for w is **4**, x is **3**, y is **3**, and z is **1**.

Summary and Additional Resources

The **solve()** function in R provides a robust and efficient mechanism for finding the unique solution to systems of linear equations. By correctly formatting the coefficient matrix (A) and the constant vector (B) using the **matrix()** function, users can quickly obtain solutions, regardless of the number of variables involved (as long as the system is non-singular).

To further enhance your skills in numerical computation and data manipulation using R, consider

exploring these related topics and tutorials:

How to perform matrix multiplication in R.
Understanding eigenvalue decomposition using R functions.
Techniques for handling sparse matrices in large datasets.