Computational Methods for Solving Equilibrium Temperature Distributions

Linear Algebra - MAS3105

Ricardo Frumento

Dr. Brian W. Curtin
Mathematics & Statistics

University of South Florida
Tampa, FL
Fall 2020

PROBLEM STATEMENT

How to find the equilibrium temperature on a large set of values using computational methods

Contents

Contents

1 Abstract

Finding the equilibrium temperature on a real life system is crucial, but it can be challenging. The number of variables and equations makes it impossible for a human to compute the final values. In this project three methods are visited: row reduction, using the inverse matrix, and LU decomposition. All methods are applied to a simple example to show the operations necessary and to a finer grid that is used to check how good approximation they are. The row reduction method showed increased accuracy when compared to the other two, but when the border temperature are not constant the LU factorization is the one with better efficiency.

2 Motivation

Everything is subject to the laws of thermodynamics. Two particular concepts that are present frequently on real life, from big civil engineering projects like damns to small computer engineering microchips, are temperature and thermal equilibrium. Temperature is related to the kinetic energies of the molecules and thermal equilibrium is defined by the zeroth law of thermodynamic, which is if systems A and B are each in thermal equilibrium with system C then systems A and B are in thermal equilibrium with each other [2]. Another concept that needs mentioning is the one of heat current, which describes the rate of heat flow from one place to another. The equation used to find it is

$$H = \frac{dQ}{dt} = kA\frac{T_H - T_C}{L}[2]$$

A problem scientists need to address is thermal stress and to do so it is required to know the temperature behaviour of every part on the object of study. One way to do it is to apply the concepts above to find the temperature on every point of the material. But it is not an easy task, the heat current equation presented is only applicable to a rod. One way to asses this is to use the mean value property[3], which states that the temperature of a point on a continuous distribution of mass that is on thermal equilibrium is given by the average temperature of all points at equal distances to the initial[1]. In Figure 1 an example is shown.

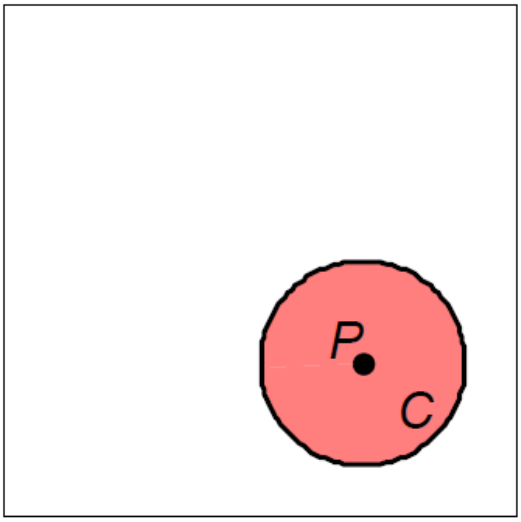

*Figure*1[1]

It is important to note that an average is $\frac{T_1-T_2}{2}$ what is similar to the equation for heat current. The problem needs further simplification because it is impossible to compute all of the points required by the property. Using a grid over the plate being studied is a good approximation. A system of equations makes it easier to find the average temperature on a simple problem but is not practical on real world dimensions or if a high precision is required. Here is where linear algebra computational methods are valuable.

3 Description and Solution

To apply three different methods, this project will use one simple problem to determine validity. In Figure 2 there is a simple set up where row reduction, the inverse matrix, and LU decomposition will be applied.

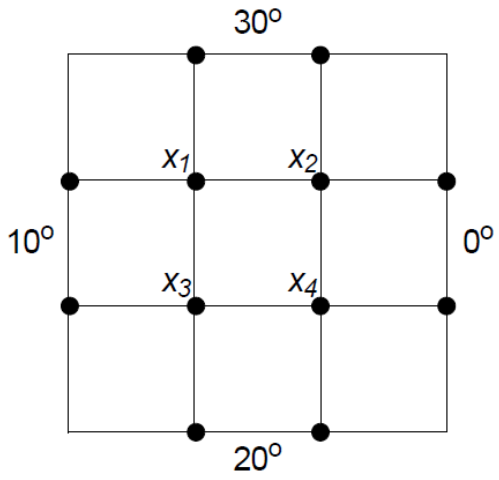

Figure 2[1]

Setting up a system of equations

$$\begin{cases} T_1 = \frac{30^\circ+10^\circ+T_2+T_3}{4} \\ T_2 = \frac{T_1+30^\circ+0^\circ+T_4}{4} \\ T_3 = \frac{10^\circ+T_1+T_4+20^\circ}{4} \\ T_4 = \frac{T_3+T_2+0^\circ+20^\circ}{4} \end{cases}$$

Rewriting

$$\begin{cases} 4T_1 = T_2 + T_3 + 40 \\ 4T_2 = T_1 + T_4 + 30 \\ 4T_3 = T_1 + T_4 + 30 \\ 4T_4 = T_2 + T_3 + 20 \end{cases}$$

This gives the matrix and the vectors

$$4\mathbf{x} = \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 40 \\ 30 \\ 30 \\ 20 \end{bmatrix}$$

Putting all variables to the left

$$\begin{cases} 4T_1 - T_2 - T_3 = 40 \\ 4T_2 - T_1 - T_4 = 30 \\ 4T_3 - T_1 - T_4 = 30 \\ 4T_4 - T_2 - T_3 = 20 \end{cases}$$

Giving the matrix and vectors

$$\mathbf{x} = \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix}, \quad Coef = \begin{bmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 40 \\ 30 \\ 30 \\ 20 \end{bmatrix}$$

3.1 Row Operations

A rectangular matrix is in echelon form when all nonzero rows are above any rows of zeros, each leading entry of a row is to the right of the leading entry of the row above and all entries below a leading entry are zeros[1]. It is possible to find a reduced echelon form, which consists of all leading entries being ones and also being the only nonzero entries of their column. This is helpful because of the uniqueness of the reduced form which states that every matrix has one and only one reduced echelon matrix. It is fairly simple to obtain the reduced echelon form of a matrix because it only relies on basic row operations, like multiplying by a scalar or adding one row to the other. As this operations create a new but equivalent matrix, it has the same reduced echelon form. All that is important because the reduced echelon form of an augmented matrix representing a linear system is the description of the solution set. If the reduced echelon form of a augmented matrix has no row of the form

$$[0 \ \ldots \ 0 \ \mathbf{b}]$$

the system is consistent and therefore has either an unique solution or a infinitely many solutions. The number of solutions is determined by the presence of free variables. On the linear system example here are the steps one should take to find the row reduced form of the augmented matrix:

$$\begin{bmatrix} 4 & -1 & -1 & 0 & 40 \\ -1 & 4 & 0 & -1 & 30 \\ -1 & 0 & 4 & -1 & 30 \\ 0 & -1 & -1 & 4 & 20 \end{bmatrix} \quad \begin{matrix} R_1 + 4R_2 \\ \\ R_3 - R_2 \\ \\ \end{matrix} \quad \rightarrow$$

$$\begin{bmatrix} 0 & 15 & -1 & -4 & 160 \\ -1 & 4 & 0 & -1 & 30 \\ 0 & -4 & 4 & 0 & 0 \\ 0 & -1 & -1 & 4 & 20 \end{bmatrix} \quad \begin{matrix} R_1 + 15R_4 \\ \\ R_3 - 4R_4 \\ \\ \end{matrix} \quad \rightarrow$$

$$\begin{bmatrix} 0 & 0 & -16 & 56 & 460 \\ -1 & 4 & 0 & -1 & 30 \\ 0 & 0 & 8 & -16 & -80 \\ 0 & -1 & -1 & 4 & 20 \end{bmatrix} \quad R_1 + 2R_3 \quad \rightarrow$$

$$\begin{bmatrix} 0 & 0 & 0 & 24 & 300 \\ -1 & 4 & 0 & -1 & 30 \\ 0 & 0 & 8 & -16 & -80 \\ 0 & -1 & -1 & 4 & 20 \end{bmatrix} \quad \begin{matrix} R_1 \textit{ moves to } R_4 \\ R_2 \textit{ moves to } R_1 \\ \\ R_4 \textit{ moves to } R_2 \\ \\ \end{matrix} \quad \rightarrow$$

$$\begin{bmatrix} -1 & 4 & 0 & -1 & 30 \\ 0 & -1 & -1 & 4 & 20 \\ 0 & 0 & 8 & -16 & -80 \\ 0 & 0 & 0 & 24 & 300 \end{bmatrix} \quad \begin{matrix} -R_1 \\ -R_2 \\ \frac{R_3}{8} \\ \frac{R_4}{24} \\ \\ \end{matrix} \quad \rightarrow$$

$$\begin{bmatrix} 1 & -4 & 0 & 1 & -30 \\ 0 & 1 & 1 & -4 & -20 \\ 0 & 0 & 1 & -2 & -10 \\ 0 & 0 & 0 & 1 & 12.5 \end{bmatrix} \quad \begin{matrix} R_1 - R_4 \\ R_2 + 4R_4 \\ R_3 + 2R_4 \\ \\ \\ \end{matrix} \quad \rightarrow$$

$$\begin{bmatrix} 1 & -4 & 0 & 0 & -42.5 \\ 0 & 1 & 1 & 0 & 30 \\ 0 & 0 & 1 & 0 & 15 \\ 0 & 0 & 0 & 1 & 12.5 \end{bmatrix} \quad R_2 - R_3 \quad \rightarrow$$

$$\begin{bmatrix} 1 & -4 & 0 & 0 & -42.5 \\ 0 & 1 & 0 & 0 & 15 \\ 0 & 0 & 1 & 0 & 15 \\ 0 & 0 & 0 & 1 & 12.5 \end{bmatrix} \quad R_1 + 4R_2 \quad \rightarrow$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 17.5 \\ 0 & 1 & 0 & 0 & 15 \\ 0 & 0 & 1 & 0 & 15 \\ 0 & 0 & 0 & 1 & 12.5 \end{bmatrix}$$

Using Matlab it is possible to use the command rref() to simplify this process (See appendix). The reduced echelon form gives the solution set for the linear system. Hence $T_1 = 17.5°, T_2 = 15°, T_3 = 15°, and T_4 = 12.5°$

3.2 Inverse Matrix

One possible way to write a linear system is as a matrix equation of the form $M\mathbf{u} = \mathbf{v}$, where M is a matrix, u and v are vectors. On the example the matrix and vectors found can be written as a matrix equation as $4\mathbf{x} = C\mathbf{x} + \mathbf{b}$. Rewriting this so it is clearer, $(4I - C)\mathbf{x} = \mathbf{b}$. Making $A = 4I - C$ the equation becomes $A\mathbf{x} = \mathbf{b}$. The solution for this equation can be found by multiplying both sides by the inverse matrix of A. It is now a matter of solving $\mathbf{x} = A^{-1}\mathbf{b}$. To find the inverse matrix there are several methods, in this project Gauss-Jordan elimination is used. It is possible to use the same row operations used on the previous method.

$$A = 4I - C = 4\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix}$$

$$[A|I] = \left[\begin{array}{cccc|cccc} 4 & -1 & -1 & 0 & 1 & 0 & 0 & 0 \\ -1 & 4 & 0 & -1 & 0 & 1 & 0 & 0 \\ -1 & 0 & 4 & -1 & 0 & 0 & 1 & 0 \\ 0 & -1 & -1 & 4 & 0 & 0 & 0 & 1 \end{array}\right] \begin{array}{l} R_1 + 4R_2 \\ \\ R_3 - R_2 \\ \\ \end{array} \rightarrow$$

$$\left[\begin{array}{cccc|cccc} 0 & 15 & -1 & -4 & 1 & 4 & 0 & 0 \\ -1 & 4 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & -4 & 4 & 0 & 0 & -1 & 1 & 0 \\ 0 & -1 & -1 & 4 & 0 & 0 & 0 & 1 \end{array}\right] \begin{array}{l} R_1 + 15R_4 \\ \\ \\ R_3 - 4R_4 \end{array} \rightarrow$$

$$\left[\begin{array}{cccc|cccc} 0 & 0 & -16 & 56 & 1 & 4 & 0 & 15 \\ -1 & 4 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 8 & -16 & 0 & -1 & 1 & -4 \\ 0 & -1 & -1 & 4 & 0 & 0 & 0 & 1 \end{array}\right] \quad R_1 + 2R_3 \quad \rightarrow$$

$$\left[\begin{array}{cccc|cccc} 0 & 0 & 0 & 24 & 1 & 2 & 2 & 7 \\ -1 & 4 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 8 & -16 & 0 & -1 & 1 & -4 \\ 0 & -1 & -1 & 4 & 0 & 0 & 0 & 1 \end{array}\right] \begin{array}{l} R_1 \textit{ moves to } R_4 \\ R_2 \textit{ moves to } R_1 \\ \\ R_4 \textit{ moves to } R_2 \end{array} \rightarrow$$

$$\left[\begin{array}{cccc|cccc} -1 & 4 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & -1 & 4 & 0 & 0 & 0 & 1 \\ 0 & 0 & 8 & -16 & 0 & -1 & 1 & -4 \\ 0 & 0 & 0 & 24 & 1 & 2 & 2 & 7 \end{array}\right] \begin{array}{c} -R_1 \\ -R_2 \\ \frac{R_3}{8} \\ \frac{R_4}{24} \end{array} \rightarrow$$

$$\left[\begin{array}{cccc|cccc} 1 & -4 & 0 & 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 1 & -4 & 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & -2 & 0 & -\frac{1}{8} & \frac{1}{8} & -\frac{1}{2} \\ 0 & 0 & 0 & 1 & \frac{1}{24} & \frac{1}{12} & \frac{1}{12} & \frac{7}{24} \end{array}\right] \begin{array}{c} R_1 - R_4 \\ R_2 + 4R_4 \\ R_3 + 2R_4 \end{array} \rightarrow$$

$$\left[\begin{array}{cccc|cccc} 1 & -4 & 0 & 0 & -\frac{1}{24} & -\frac{13}{12} & -\frac{1}{12} & -\frac{7}{24} \\ 0 & 1 & 1 & 0 & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \\ 0 & 0 & 1 & 0 & \frac{1}{12} & \frac{1}{24} & \frac{7}{24} & \frac{1}{12} \\ 0 & 0 & 0 & 1 & \frac{1}{24} & \frac{1}{12} & \frac{1}{12} & \frac{7}{24} \end{array}\right] R_2 - R_3 \rightarrow$$

$$\left[\begin{array}{cccc|cccc} 1 & -4 & 0 & 0 & -\frac{1}{24} & -\frac{13}{12} & -\frac{1}{12} & -\frac{7}{24} \\ 0 & 1 & 0 & 0 & \frac{1}{12} & \frac{7}{24} & \frac{1}{24} & \frac{1}{12} \\ 0 & 0 & 1 & 0 & \frac{1}{12} & \frac{1}{24} & \frac{7}{24} & \frac{1}{12} \\ 0 & 0 & 0 & 1 & \frac{1}{24} & \frac{1}{12} & \frac{1}{12} & \frac{7}{24} \end{array}\right] R_1 + 4R_2 \rightarrow$$

$$\left[\begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & \frac{7}{24} & \frac{1}{12} & \frac{1}{12} & \frac{1}{24} \\ 0 & 1 & 0 & 0 & \frac{1}{12} & \frac{7}{24} & \frac{1}{24} & \frac{1}{12} \\ 0 & 0 & 1 & 0 & \frac{1}{12} & \frac{1}{24} & \frac{7}{24} & \frac{1}{12} \\ 0 & 0 & 0 & 1 & \frac{1}{24} & \frac{1}{12} & \frac{1}{12} & \frac{7}{24} \end{array}\right]$$

After this process the inverted matrix is applied to the matrix equation

$$\mathbf{x} = A^{-1}\mathbf{b}$$

$$\begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} = \begin{bmatrix} \frac{7}{24} & \frac{1}{12} & \frac{1}{12} & \frac{1}{24} \\ \frac{1}{12} & \frac{7}{24} & \frac{1}{24} & \frac{1}{12} \\ \frac{1}{12} & \frac{1}{24} & \frac{7}{24} & \frac{1}{12} \\ \frac{1}{24} & \frac{1}{12} & \frac{1}{12} & \frac{7}{24} \end{bmatrix} \begin{bmatrix} 40 \\ 30 \\ 30 \\ 20 \end{bmatrix} = \begin{bmatrix} 17.5 \\ 15 \\ 15 \\ 12.5 \end{bmatrix}$$

One way to simplify the process is to use the MAT Lab built in function inv() (See Appendix). This are the same equilibrium temperature found for the method using row operations, $T_1 = 17.5°, T_2 = 15°, T_3 = 15°$, and $T_4 = 12.5°$

3.3 LU Decomposition

The two previous methods are good in solving the proposed problem. But one downside both have is that, if the initial values change it is necessary to start over the whole process. The LU factorization is a clever way to solve

multiple problems with the same coefficient matrix. Given a matrix $A = LU$, where L and U are triangular matrices, the equation $A\mathbf{x} = \mathbf{b}$ can be rewritten as $L(U\mathbf{x}) = \mathbf{b}$. Writing $\mathbf{y}$ for $U\mathbf{x}$ it is only a matter of solving the pair of equations:

$$L\mathbf{y} = \mathbf{b}$$
$$U\mathbf{x} = \mathbf{y}$$

To find the matrices it is necessary to find the echelon form of the upper triangular and then use the steps to find the lower triangular. Using the same coefficient matrix as before

$$C = \begin{bmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix} \quad \begin{matrix} R_2 + \frac{1}{4}R_1 \\ R_3 + \frac{1}{4}R_1 \end{matrix} \quad \rightarrow$$

$$\begin{bmatrix} 4 & -1 & -1 & 0 \\ 0 & \frac{15}{4} & -\frac{1}{4} & -1 \\ 0 & -\frac{1}{4} & \frac{15}{4} & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix} \quad \begin{matrix} R_3 + \frac{1}{15}R_2 \\ R_4 + \frac{4}{15}R_2 \end{matrix} \quad \rightarrow$$

$$\begin{bmatrix} 4 & -1 & -1 & 0 \\ 0 & \frac{15}{4} & -\frac{1}{4} & -1 \\ 0 & 0 & \frac{56}{15} & -\frac{16}{15} \\ 0 & 0 & -\frac{16}{15} & \frac{56}{15} \end{bmatrix} \quad R_4 + \frac{16}{56}R_3 \quad \rightarrow \quad U = \begin{bmatrix} 4 & -1 & -1 & 0 \\ 0 & \frac{15}{4} & -\frac{1}{4} & -1 \\ 0 & 0 & \frac{56}{15} & -\frac{16}{15} \\ 0 & 0 & 0 & \frac{24}{7} \end{bmatrix}$$

Once the upper triangular matrix is obtained, the lower is found observing that $E_p \dots E_1 A = U$ then $A = (E_p \dots E_1)^{-1}U = LU$ resulting in $L = (E_p \dots E_1)^{-1}$ So to find L it is necessary to pay attention to the columns in C during the process of reducing. The method is as shown below

$$L = \begin{bmatrix} 4.\frac{1}{4} & 0 & 0 & 0 \\ -1.\frac{1}{4} & \frac{15}{4}\cdot\frac{4}{15} & 0 & 0 \\ -1.\frac{1}{4} & -\frac{1}{4}\cdot\frac{4}{15} & \frac{56}{15}\cdot\frac{15}{56} & 0 \\ 0 & -1.\frac{4}{15} & -\frac{16}{15}\cdot\frac{15}{56} & \frac{216}{63}\cdot\frac{63}{216} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -\frac{1}{4} & 1 & 0 & 0 \\ -\frac{1}{4} & -\frac{1}{15} & 1 & 0 \\ 0 & -\frac{4}{15} & -\frac{16}{56} & 1 \end{bmatrix}$$

Now that both the triangular matrices are found the solution of the pair of equations is easy to compute

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ -\frac{1}{4} & 1 & 0 & 0 \\ -\frac{1}{4} & -\frac{1}{15} & 1 & 0 \\ 0 & -\frac{4}{15} & -\frac{16}{56} & 1 \end{bmatrix} \mathbf{y} = \mathbf{b} = \begin{bmatrix} 40 \\ 30 \\ 30 \\ 20 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 40 \\ -\frac{1}{4} & 1 & 0 & 0 & 30 \\ -\frac{1}{4} & -\frac{1}{15} & 1 & 0 & 30 \\ 0 & -\frac{4}{15} & -\frac{16}{56} & 1 & 20 \end{bmatrix} \quad \begin{matrix} R_2 + \frac{1}{4}R_1 \\ R_3 + \frac{1}{4}R_1 \end{matrix} \quad \rightarrow$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 40 \\ 0 & 1 & 0 & 0 & 40 \\ 0 & -\frac{1}{15} & 1 & 0 & 40 \\ 0 & -\frac{4}{15} & -\frac{16}{56} & 1 & 20 \end{bmatrix} \quad \begin{matrix} R_3 + \frac{1}{15}R_2 \\ R_4 + \frac{4}{15}R_2 \end{matrix} \quad \rightarrow$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 40 \\ 0 & 1 & 0 & 0 & 40 \\ 0 & 0 & 1 & 0 & \frac{128}{3} \\ 0 & 0 & -\frac{16}{56} & 1 & \frac{92}{3} \end{bmatrix} \quad R_4 + \frac{16}{56}R_3 \quad \rightarrow \quad \begin{bmatrix} 1 & 0 & 0 & 0 & 40 \\ 0 & 1 & 0 & 0 & 40 \\ 0 & 0 & 1 & 0 & \frac{128}{3} \\ 0 & 0 & 0 & 1 & \frac{300}{7} \end{bmatrix}$$

And then

$$\begin{bmatrix} 4 & -1 & -1 & 0 \\ 0 & \frac{15}{4} & -\frac{1}{4} & -1 \\ 0 & 0 & \frac{56}{15} & -\frac{16}{15} \\ 0 & 0 & 0 & \frac{24}{7} \end{bmatrix} \text{x} = \text{y} = \begin{bmatrix} 40 \\ 40 \\ \frac{128}{3} \\ \frac{300}{7} \end{bmatrix}$$

$$\begin{bmatrix} 4 & -1 & -1 & 0 & 40 \\ 0 & \frac{15}{4} & -\frac{1}{4} & -1 & 40 \\ 0 & 0 & \frac{56}{15} & -\frac{16}{15} & \frac{128}{3} \\ 0 & 0 & 0 & \frac{24}{7} & \frac{300}{7} \end{bmatrix} \quad \frac{7}{24}R_4 \quad \rightarrow$$

$$\begin{bmatrix} 4 & -1 & -1 & 0 & 40 \\ 0 & \frac{15}{4} & -\frac{1}{4} & -1 & 40 \\ 0 & 0 & \frac{56}{15} & -\frac{16}{15} & \frac{128}{3} \\ 0 & 0 & 0 & 1 & \frac{25}{2} \end{bmatrix} \quad \begin{matrix} R_2 + R_4 \\ R_3 + \frac{16}{15}R_4 \end{matrix} \quad \rightarrow \quad \begin{bmatrix} 4 & -1 & -1 & 0 & 40 \\ 0 & \frac{15}{4} & -\frac{1}{4} & 0 & \frac{105}{2} \\ 0 & 0 & \frac{56}{15} & 0 & 56 \\ 0 & 0 & 0 & 1 & \frac{25}{2} \end{bmatrix} \quad \frac{15}{56}R_3 \quad \rightarrow$$

$$\begin{bmatrix} 4 & -1 & -1 & 0 & 40 \\ 0 & \frac{15}{4} & -\frac{1}{4} & 0 & \frac{105}{2} \\ 0 & 0 & 1 & 0 & 15 \\ 0 & 0 & 0 & 1 & \frac{25}{2} \end{bmatrix} \quad \begin{matrix} R_1 + R_3 \\ R_2 + \frac{1}{4}R_3 \end{matrix} \quad \rightarrow$$

$$\begin{bmatrix} 4 & -1 & 0 & 0 & 55 \\ 0 & \frac{15}{4} & 0 & 0 & \frac{225}{4} \\ 0 & 0 & 1 & 0 & 15 \\ 0 & 0 & 0 & 1 & \frac{25}{2} \end{bmatrix} \quad \frac{4}{15}R_2 \quad \rightarrow \quad \begin{bmatrix} 4 & -1 & 0 & 0 & 55 \\ 0 & 1 & 0 & 0 & 15 \\ 0 & 0 & 1 & 0 & 15 \\ 0 & 0 & 0 & 1 & \frac{25}{2} \end{bmatrix} \quad R_1 + R_2$$

$$\rightarrow \quad \begin{bmatrix} 4 & 0 & 0 & 0 & 70 \\ 0 & 1 & 0 & 0 & 15 \\ 0 & 0 & 1 & 0 & 15 \\ 0 & 0 & 0 & 1 & \frac{25}{2} \end{bmatrix} \quad \frac{1}{4}R_1 \quad \rightarrow \quad \begin{bmatrix} 1 & 0 & 0 & 0 & \frac{35}{2} \\ 0 & 1 & 0 & 0 & 15 \\ 0 & 0 & 1 & 0 & 15 \\ 0 & 0 & 0 & 1 & \frac{25}{2} \end{bmatrix}$$

It is possible to use the MATLab built in function lu() to find the two triangular matrices (See Appendix). This are the same values the previous methods found $T_1 = 17.5°$, $T_2 = 15°$, $T_3 = 15°$, and $T_4 = 12.5°$, but there is a benefit to the LU factorization. That is to reduce the computations when the initial conditions change. In this example, if the temperatures of the borders change

3.4 Different Border Temperature

If the temperature of the upper border changes to 50° the process must be done all over again for the row operations and the inverse matrix methods. But for the LU decomposition it is only a matter of solving the new pair of equations. As the solution is obtained with the same process already presented for this part only the Matlab computations will be shown (See Appendix).

$$\begin{cases} T_1 = \frac{50°+10°+T_2+T_3}{4} \\ T_2 = \frac{T_1+50°+0°+T_4}{4} \\ T_3 = \frac{10°+T_1+T_4+20°}{4} \\ T_4 = \frac{T_3+T_2+0°+20°}{4} \end{cases}$$

$$\mathbf{x} = \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix}, \quad Coef = \begin{bmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 60 \\ 50 \\ 30 \\ 20 \end{bmatrix}$$

3.5 A Finer Grid

For further analysis on the approximation given by the methods, all three will be applied on the same plate but now with 25 grid points. The data is pulled from an external source. Following are the results as the computations are hidden (See Appendix).

```
disp([xred xinv xlu])
```

```
  19.373737373737374  19.373737373737374  19.373737373737370
  22.401315789473685  22.401320901320897  22.401320901320897
  22.655737704918032  22.655788655788651  22.655788655788644
  20.663934426229510  20.663947163947157  20.663947163947153
  15.000000000000000  14.999999999999998  15.000000000000000
  15.093750000000000  15.093628593628594  15.093628593628594
  17.575757575757574  17.575757575757571  17.575757575757571
  17.557692307692307  17.557886557886555  17.557886557886555
  15.000000000000000  14.999999999999995  14.999999999999995
   9.336065573770492   9.336052836052833   9.336052836052835
```

```
13.425000000000001  13.425019425019421  13.425019425019423
15.250000000000000  15.250194250194241  15.250194250194243
15.000000000000000  14.999999999999995  14.999999999999995
12.442307692307692  12.442113442113440  12.442113442113438
 7.344262295081967   7.344211344211344   7.344211344211343
13.356321839080460  13.356254856254854  13.356254856254855
15.000000000000000  14.999999999999991  14.999999999999991
14.750000000000000  14.749805749805745  14.749805749805748
12.424242424242424  12.424242424242424  12.424242424242419
 7.598684210526316   7.598679098679098   7.598679098679098
15.000000000000000  14.999999999999995  15.000000000000000
16.643678160919539  16.643745143745143  16.643745143745143
16.574999999999999  16.574980574980572  16.574980574980572
14.906250000000000  14.906371406371404  14.906371406371402
10.626262626262626  10.626262626262625  10.626262626262625
```

4 Discussions

For this project four aspects will be discussed, how many elementary row operations were necessary, when the initial conditions changed how many were necessary, the accuracy of the computational method and how the answers compare when there are more variables and equations. Using row reduction required 38 computations, the inverse required 60, and the LU decomposition 42. This already shows how less efficient the inverse method is, it requires almost a third more operations than the other two and this number only tends to grow with the increase of variables and equations. The tie breaker between the two remaining methods is the amount of operations required after the change of the initial values. The row reduction method takes the same 38 operations but the LU factorization takes only 24, that is due to the ability of using the same lower and upper triangular matrices found for the first part as neither were found using the initial values. This difference tends to show on bigger, more realistic systems, that the temperatures change constantly and the operation is required to be done more frequently.

The third characteristic evaluated is the accuracy and that is found by the methods that required successive rounding, in this case the inverse is also the worst of the three because it first rounds values to find the inverse and then rounds the multiplication result $A\mathbf{x} = \mathbf{b}$ to $\mathbf{x} = A^{-1}\mathbf{b}$. When comparing the remaining two methods there are no matrix equations to be solved but it is still possible to compare them, as for the row reduction method there is only one augmented matrix to be reduced, for the LU decomposition there are two, of course they are triangular matrices but that influences on the amount of independent operations and not the rounding done when two operations are done in sequence, for example. Finally, to discuss how the answers compare it is important to use the results found on the finer grid. Below is the comparison

of the temperatures found, first column represents the values found by using the inverse subtracted from the ones found by row reducing, second column show the values LU decomposition found subtracted from the ones row reduced found, and third column is the values LU decomposition found subtracted from the ones finding the inverse found.

```
disp([(xred-xinv) (xred-xlu) (xinv-xlu)])
```

```
 1.0e-03 *

                 0   0.000000000003553   0.000000000003553
-0.005111847212191  -0.005111847212191                   0
-0.050950870619459  -0.050950870612354   0.000000000007105
-0.012737717646871  -0.012737717643319   0.000000000003553
 0.000000000001776                   0  -0.000000000001776
 0.121406371405897   0.121406371405897                   0
 0.000000000003553   0.000000000003553                   0
-0.194250194248013  -0.194250194248013                   0
 0.000000000005329   0.000000000005329                   0
 0.012737717659306   0.012737717657529  -0.000000000001776
-0.019425019420538  -0.019425019422314  -0.000000000001776
-0.194250194240908  -0.194250194242684  -0.000000000001776
 0.000000000005329   0.000000000005329                   0
 0.194250194251566   0.194250194253343   0.000000000001776
 0.050950870623012   0.050950870624789   0.000000000001776
 0.066982825606132   0.066982825604356  -0.000000000001776
 0.000000000008882   0.000000000008882                   0
 0.194250194255119   0.194250194251566  -0.000000000003553
                 0   0.000000000005329   0.000000000005329
 0.005111847218409   0.005111847217520  -0.000000000000888
 0.000000000005329                   0  -0.000000000005329
-0.066982825604356  -0.066982825604356                   0
 0.019425019427644   0.019425019427644                   0
-0.121406371404120  -0.121406371402344   0.000000000001776
 0.000000000001776   0.000000000001776                   0
```

This table shows values that differ on the forth decimal plate and if precision is a requirement of the project this might be problematic. As discussed, the two methods that execute more operations in sequence have closer numbers and that is due to the error being carried through sequential computations.

5 Conclusions

From the three methods used in this project two stand out, the row reduction and the LU factorization. The first one would be recommended for one

time computations due to the increased accuracy. The second one would be recommended for continuous analysis that require several computations with different initial conditions. MATLab has shown to be a powerful tool to achieve such computations, the examples used on this project were also solved using built in functions and are presented on the Appendix. As the methods did not use any equations based on thermodynamics all of them can be used to solve other systems of linear equations and the same observations about accuracy and efficiency apply.

References

[1] David C. Lay, Steven R. Lay, and Judi J. McDonald *Linear Algebra and Its Applications*. Pearson, 2020.

[2] Hugh D. Young, Roger A. Freedman *University Physics* Pearson, 2020

[3] *Harmonic function* (2020, September 26). In Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Harmonic_function

6 Appendix: MATLab Computations

Row Operations

```
Coef = [4 -1 -1 0; -1 4 0 -1; -1 0 4 -1; 0 -1 -1 4]
```

```
Coef = 4x4
     4    -1    -1     0
    -1     4     0    -1
    -1     0     4    -1
     0    -1    -1     4
```

```
b = [40 30 30 20]'
```

```
b = 4x1
    40
    30
    30
    20
```

```
Aug = [Coef b]
```

```
Aug = 4x5
     4    -1    -1     0    40
    -1     4     0    -1    30
    -1     0     4    -1    30
     0    -1    -1     4    20
```

```
Reduced = rref(Aug)
```

```
Reduced = 4x5
    1.0000         0         0         0   17.5000
         0    1.0000         0         0   15.0000
         0         0    1.0000         0   15.0000
         0         0         0    1.0000   12.5000
```

Inverse Matrix

```
C = [0 1 1 0; 1 0 0 1; 1 0 0 1; 0 1 1 0]
```

```
C = 4x4
     0     1     1     0
     1     0     0     1
     1     0     0     1
     0     1     1     0
```

```
b = [40 30 30 20]'
```

```
b = 4x1
    40
    30
    30
    20
```

```
I = eye(4)
```

```
I = 4x4
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
```

```
A = 4*I-C
```

```
A = 4x4
     4    -1    -1     0
    -1     4     0    -1
    -1     0     4    -1
     0    -1    -1     4
```

```
Ainv = inv(A)
```

```
Ainv = 4x4
    0.2917    0.0833    0.0833    0.0417
    0.0833    0.2917    0.0417    0.0833
    0.0833    0.0417    0.2917    0.0833
    0.0417    0.0833    0.0833    0.2917
```

```
answer = Ainv*b
```

```
answer = 4x1
   17.5000
   15.0000
   15.0000
   12.5000
```

LU Decomposition

```
Coef = [4 -1 -1 0; -1 4 0 -1; -1 0 4 -1; 0 -1 -1 4]
```

```
Coef = 4x4
     4    -1    -1     0
    -1     4     0    -1
    -1     0     4    -1
     0    -1    -1     4
```

```
b = [40 30 30 20]'
```

```
b = 4x1
    40
    30
    30
    20
```

```
[L U] = lu(Coef)
```

```
L = 4x4
    1.0000         0         0         0
   -0.2500    1.0000         0         0
   -0.2500   -0.0667    1.0000         0
         0   -0.2667   -0.2857    1.0000

U = 4x4
    4.0000   -1.0000   -1.0000         0
         0    3.7500   -0.2500   -1.0000
         0         0    3.7333   -1.0667
         0         0         0    3.4286
```

```
AugL = [L b]
```

```
AugL = 4x5
    1.0000         0         0         0   40.0000
   -0.2500    1.0000         0         0   30.0000
   -0.2500   -0.0667    1.0000         0   30.0000
         0   -0.2667   -0.2857    1.0000   20.0000
```

```
M = rref(AugL)
```

```
M = 4x5
    1.0000         0         0         0   40.0000
         0    1.0000         0         0   40.0000
         0         0    1.0000         0   42.6667
         0         0         0    1.0000   42.8571
```

```
y = [M(:,5)]
```

```
y = 4x1
   40.0000
   40.0000
   42.6667
   42.8571
```

```
AugU = [U y]
```

```
AugU = 4x5
    4.0000   -1.0000   -1.0000         0   40.0000
         0    3.7500   -0.2500   -1.0000   40.0000
         0         0    3.7333   -1.0667   42.6667
         0         0         0    3.4286   42.8571
```

```
M = rref(AugU)
```

```
M = 4x5
    1.0000         0         0         0   17.5000
         0    1.0000         0         0   15.0000
         0         0    1.0000         0   15.0000
         0         0         0    1.0000   12.5000
```

```
x = [M(:,5)]
```

```
x = 4x1
   17.5000
   15.0000
   15.0000
   12.5000
```

Different Border Temperature
Row Operations:

```
Coef = [4 -1 -1 0; -1 4 0 -1; -1 0 4 -1; 0 -1 -1 4]
```

```
Coef = 4x4
     4    -1    -1     0
    -1     4     0    -1
    -1     0     4    -1
     0    -1    -1     4
```

```
b = [60 50 30 20]'
```

```
b = 4x1
    60
    50
    30
    20
```

```
Aug = [Coef b]
```

```
Aug = 4x5
     4    -1    -1     0    60
    -1     4     0    -1    50
    -1     0     4    -1    30
     0    -1    -1     4    20
```

```
Reduced = rref(Aug)
```

```
Reduced = 4x5
    1.0000         0         0         0   25.0000
         0    1.0000         0         0   22.5000
         0         0    1.0000         0   17.5000
         0         0         0    1.0000   15.0000
```

```
x1 = Reduced(:,5)
```

```
x1 = 4x1
   25.0000
   22.5000
   17.5000
   15.0000
```

Inverse Matrix:
$A = 4I - C$

```
A = Coef
```

```
A = 4x4
     4    -1    -1     0
    -1     4     0    -1
    -1     0     4    -1
     0    -1    -1     4
```

```
Ainv = inv(A)
```

```
Ainv = 4x4
    0.2917    0.0833    0.0833    0.0417
    0.0833    0.2917    0.0417    0.0833
    0.0833    0.0417    0.2917    0.0833
    0.0417    0.0833    0.0833    0.2917
```

```
x2 = Ainv*b
```

```
x2 = 4x1
   25.0000
   22.5000
   17.5000
   15.0000
```

LU Decomposition

```
AugL = [L b]
```

```
AugL = 4x5
    1.0000         0         0         0   60.0000
   -0.2500    1.0000         0         0   50.0000
   -0.2500   -0.0667    1.0000         0   30.0000
         0   -0.2667   -0.2857    1.0000   20.0000
```

```
M = rref(AugL)
```

```
M = 4x5
    1.0000         0         0         0   60.0000
         0    1.0000         0         0   65.0000
         0         0    1.0000         0   49.3333
         0         0         0    1.0000   51.4286
```

```
y = M(:,5)
```

```
y = 4x1
   60.0000
   65.0000
   49.3333
   51.4286
```

```
AugU = [U y]
```

```
AugU = 4x5
    4.0000   -1.0000   -1.0000         0   60.0000
         0    3.7500   -0.2500   -1.0000   65.0000
         0         0    3.7333   -1.0667   49.3333
         0         0         0    3.4286   51.4286
```

```
M = rref(AugU)
```

```
M = 4x5
    1.0000         0         0         0   25.0000
         0    1.0000         0         0   22.5000
         0         0    1.0000         0   17.5000
         0         0         0    1.0000   15.0000
```

```
x3 = M(:,5)
```

```
x3 = 4x1
   25.0000
   22.5000
   17.5000
   15.0000
```

```
X = [x1 x2 x3]
```

```
X = 4x3
   25.0000   25.0000   25.0000
   22.5000   22.5000   22.5000
   17.5000   17.5000   17.5000
   15.0000   15.0000   15.0000
```

A Finer Grid

```
format long
temperature
```

```
C = 25x25
     0     1     0     0     0     1     0     0     0     0     0     0     0     0
     1     0     1     0     0     0     1     0     0     0     0     0     0     0
     0     1     0     1     0     0     0     1     0     0     0     0     0     0
     0     0     1     0     1     0     0     0     1     0     0     0     0     0
     0     0     0     1     0     0     0     0     0     1     0     0     0     0
     1     0     0     0     0     0     1     0     0     0     1     0     0     0
     0     1     0     0     0     1     0     1     0     0     0     1     0     0
     0     0     1     0     0     0     1     0     1     0     0     0     1     0
     0     0     0     1     0     0     0     1     0     1     0     0     0     1
     0     0     0     0     1     0     0     0     1     0     0     0     0     0
```

```
b = 25x1
    40
    30
    30
    30
    30
    10
     0
     0
     0
     0
```

Row Operations

```
A = 4*eye(25)-C
```

```
A = 25x25
     4    -1     0     0     0    -1     0     0     0     0     0     0     0     0
    -1     4    -1     0     0     0    -1     0     0     0     0     0     0     0
     0    -1     4    -1     0     0     0    -1     0     0     0     0     0     0
     0     0    -1     4    -1     0     0     0    -1     0     0     0     0     0
     0     0     0    -1     4     0     0     0     0    -1     0     0     0     0
    -1     0     0     0     0     4    -1     0     0     0    -1     0     0     0
     0    -1     0     0     0    -1     4    -1     0     0     0    -1     0     0
     0     0    -1     0     0     0    -1     4    -1     0     0     0    -1     0
     0     0     0    -1     0     0     0    -1     4    -1     0     0     0    -1
     0     0     0     0    -1     0     0     0    -1     4     0     0     0     0    -
```

```
Aug = [A b]
```

```
Aug = 25x26
     4    -1     0     0     0    -1     0     0     0     0     0     0     0     0
    -1     4    -1     0     0     0    -1     0     0     0     0     0     0     0
     0    -1     4    -1     0     0     0    -1     0     0     0     0     0     0
     0     0    -1     4    -1     0     0     0    -1     0     0     0     0     0
     0     0     0    -1     4     0     0     0     0    -1     0     0     0     0
    -1     0     0     0     0     4    -1     0     0     0    -1     0     0     0
     0    -1     0     0     0    -1     4    -1     0     0     0    -1     0     0
     0     0    -1     0     0     0    -1     4    -1     0     0     0    -1     0
     0     0     0    -1     0     0     0    -1     4    -1     0     0     0    -1
     0     0     0     0    -1     0     0     0    -1     4     0     0     0     0    -
```

```
Reduced = rref(Aug)
```

```
Reduced = 25x26
   1.000000000000000                   0                   0                   0
                   0   1.000000000000000                   0                   0
                   0                   0   1.000000000000000                   0
                   0                   0                   0   1.000000000000000
                   0                   0                   0                   0   1.0000
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
```

```
xred = Reduced(:,26)
```

```
xred = 25x1
  19.373737373737374
  22.401315789473685
  22.655737704918032
  20.663934426229510
  15.000000000000000
  15.093750000000000
  17.575757575757574
  17.557692307692307
  15.000000000000000
   9.336065573770492
```

Inverse Matrix

```
Ainv = inv(A)
```

```
Ainv = 25x25
   0.301738539238539   0.103477078477078   0.040200077700078   0.016860916860917   0.0064
   0.103477078477078   0.341938616938617   0.120337995337995   0.046610334110334   0.0168
   0.040200077700078   0.120337995337995   0.348348873348873   0.120337995337995   0.0402
   0.016860916860917   0.046610334110334   0.120337995337995   0.341938616938617   0.1034
   0.006410256410256   0.016860916860917   0.040200077700078   0.103477078477078   0.3017
   0.103477078477078   0.071969696969697   0.040462315462315   0.020833333333333   0.0087
   0.071969696969697   0.143939393939394   0.092803030303030   0.049242424242424   0.0208
   0.040462315462315   0.092803030303030   0.152719502719503   0.092803030303030   0.0404
   0.020833333333333   0.049242424242424   0.092803030303030   0.143939393939394   0.0719
   0.008780108780109   0.020833333333333   0.040462315462315   0.071969696969697   0.1034
```

```
xinv = Ainv*b
```

```
xinv = 25x1
  19.373737373737374
  22.401320901320897
  22.655788655788651
  20.663947163947157
  14.999999999999998
  15.093628593628594
  17.575757575757571
  17.557886557886555
  14.999999999999995
   9.336052836052833
```

LU Decomposition

```
[L U] = lu(A)
```

```
L = 25x25
   1.000000000000000                   0                   0                   0
  -0.250000000000000   1.000000000000000                   0                   0
                   0  -0.266666666666667   1.000000000000000                   0
                   0                   0  -0.267857142857143   1.000000000000000
                   0                   0                   0  -0.267942583732057
  -0.250000000000000  -0.066666666666667  -0.017857142857143  -0.004784688995215  -0.0012
                   0  -0.266666666666667  -0.071428571428571  -0.019138755980861  -0.0051
                   0                   0  -0.267857142857143  -0.071770334928230  -0.0192
```

```
                  0                   0                   0  -0.267942583732057  -0.0717
                  0                   0                   0                   0  -0.2679

U = 25x25
   4.000000000000000  -1.000000000000000                   0                   0
                  0   3.750000000000000  -1.000000000000000                   0
                  0                   0   3.733333333333333  -1.000000000000000
                  0                   0                   0   3.732142857142857  -1.0000
                  0                   0                   0                   0   3.7320
                  0                   0                   0                   0
                  0                   0                   0                   0
                  0                   0                   0                   0
                  0                   0                   0                   0
                  0                   0                   0                   0
```

```
AugL = [L b]
```

```
AugL = 25x26
   1.000000000000000                   0                   0                   0
  -0.250000000000000   1.000000000000000                   0                   0
                  0  -0.266666666666667   1.000000000000000                   0
                  0                   0  -0.267857142857143   1.000000000000000
                  0                   0                   0  -0.267942583732057   1.0000
  -0.250000000000000  -0.066666666666667  -0.017857142857143  -0.004784688995215  -0.0012
                  0  -0.266666666666667  -0.071428571428571  -0.019138755980861  -0.0051
                  0                   0  -0.267857142857143  -0.071770334928230  -0.0192
                  0                   0                   0  -0.267942583732057  -0.0717
                  0                   0                   0                   0  -0.2679
```

```
M = rref(AugL)
```

```
M = 25x26
   1.000000000000000                   0                   0                   0
                  0   1.000000000000000                   0                   0
                  0                   0   1.000000000000000                   0
                  0                   0                   0   1.000000000000000
                  0                   0                   0                   0   1.0000
                  0                   0                   0                   0
                  0                   0                   0                   0
                  0                   0                   0                   0
                  0                   0                   0                   0
                  0                   0                   0                   0
```

```
y = M(:,26)
```

```
y = 25x1
  40.000000000000000
  40.000000000000000
  40.666666666666664
  40.892857142857139
  40.956937799043061
  23.641025641025639
  21.353486774304361
  21.525423728813557
  20.996794199598522
  17.883710987601539
```

```
AugU = [U y]
```

```
AugU = 25x26
   4.000000000000000  -1.000000000000000                   0                   0
                   0   3.750000000000000  -1.000000000000000                   0
                   0                   0   3.733333333333333  -1.000000000000000
                   0                   0                   0   3.732142857142857  -1.0000
                   0                   0                   0                   0   3.7320
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
```

```
M = rref(AugU)
```

```
M = 25x26
   1.000000000000000                   0                   0                   0
                   0   1.000000000000000                   0                   0
                   0                   0   1.000000000000000                   0
                   0                   0                   0   1.000000000000000
                   0                   0                   0                   0   1.0000
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
                   0                   0                   0                   0
```

```
xlu = M(:,26)
```

```
xlu =
  19.373737373737370
  22.401320901320897
  22.655788655788644
  20.663947163947153
  15.000000000000000
  15.093628593628594
  17.575757575757571
  17.557886557886555
  14.999999999999995
   9.336052836052835
```

```
disp([xred xinv xlu])
```

```
  19.373737373737374  19.373737373737374  19.373737373737370
  22.401315789473685  22.401320901320897  22.401320901320897
  22.655737704918032  22.655788655788651  22.655788655788644
  20.663934426229510  20.663947163947157  20.663947163947153
  15.000000000000000  14.999999999999998  15.000000000000000
  15.093750000000000  15.093628593628594  15.093628593628594
  17.575757575757574  17.575757575757571  17.575757575757571
  17.557692307692307  17.557886557886555  17.557886557886555
  15.000000000000000  14.999999999999995  14.999999999999995
   9.336065573770492   9.336052836052833   9.336052836052835
  13.425000000000001  13.425019425019421  13.425019425019423
  15.250000000000000  15.250194250194241  15.250194250194243
  15.000000000000000  14.999999999999995  14.999999999999995
  12.442307692307692  12.442113442113440  12.442113442113438
   7.344262295081967   7.344211344211344   7.344211344211343
  13.356321839080460  13.356254856254854  13.356254856254855
  15.000000000000000  14.999999999999991  14.999999999999991
  14.750000000000000  14.749805749805745  14.749805749805748
  12.424242424242424  12.424242424242424  12.424242424242419
   7.598684210526316   7.598679098679098   7.598679098679098
  15.000000000000000  14.999999999999995  15.000000000000000
  16.643678160919539  16.643745143745143  16.643745143745143
  16.574999999999999  16.574980574980572  16.574980574980572
  14.906250000000000  14.906371406371404  14.906371406371402
  10.626262626262626  10.626262626262625  10.626262626262625
```