The Application of Cholesky Factorization of *n* x *n* Matrices in Solving Numerical Methods as Stipulated by Chemical Engineering Problems

by

Sam Ly, Xing (Jim) Xiang, and Alfred Joshua Aquino

An Exploratory Assignment
presented to Dr. Meunier
in fulfillment of the
(possible) 7.5% Grade Requirement
for CHE 322
in
Chemical Engineering

Waterloo, Ontario, Canada, 2018

Authors' Declaration

This exploratory assignment consists of material all of which we authored or co-authored. Any material outside the immediate scope of the topic at hand or material in which was retrieved from tertiary sources have been credited accordingly. This is a true copy of the exploratory assignment, including any required final revisions, as required.

We understand that our exploratory assignment may be made electronically available via Learn.

Statement of Contributions

We would like to thank Dr. Sarah M. Meunier for providing such an amazing opportunity to learn and further our knowledge of numerical methods and for her outstanding lecturing capabilities in her numerical methods lecture series, specifically for making methods easy to digest, understand, and implement. We would also like to thank the University of Waterloo for letting us attend this institution where it fosters and incubates individuals for success. *Ideas Start Here*.

Table of Contents

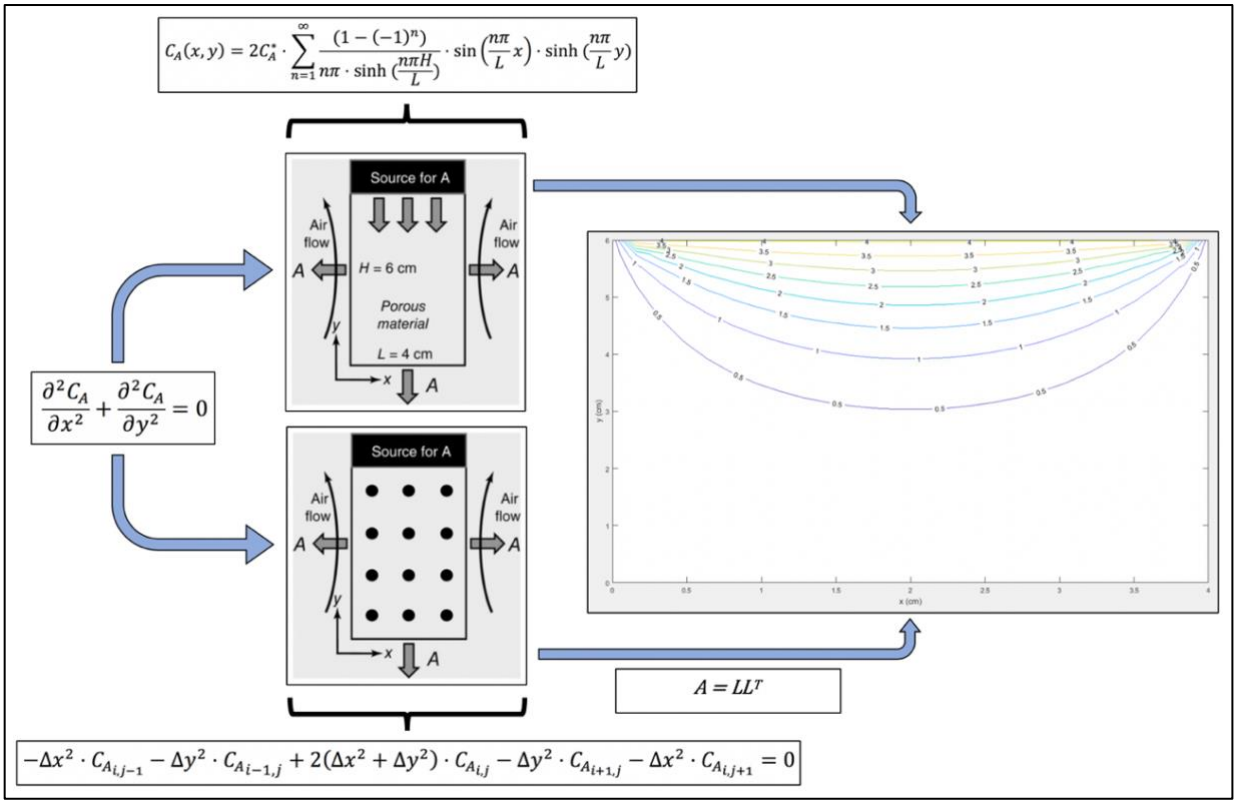
List of Figures

List of Appendices

Abstract

Documentation was developed for the Cholesky factorization numerical method. The method was applied to a two-dimensional mass transport problem with its analytical and numerical solutions compared via a contour plot displaying concentration profile constructed on MATLAB. The concentration profiles obtained numerically and analytically were identical. Further, the Cholesky factorization had faster runtime than LU factorization by approximately 0.50 seconds. The method was advantageous in its single determination of a constituent matrix *L*, such that the factorization of a system, *A*, was of the form $A = LL^T$. The drawback of using this method was its specific application to symmetric and positive-definite systems of equations. It was recommended to ensure that a matrix was symmetric and positive-definite before implementing Cholesky factorization; furthermore, error analysis for the method and its comparison with LU factorization for systems of equations that are not sparse can be conducted for future studies.

Graphical Abstract



Introduction

In linear algebra, complex matrix operations are often characterized by instability and inefficient solvability. Such complications are overcome via matrix factorization, which involves reducing a matrix into a product of matrices for an efficient solution. Fortunately, a variety of factorization methods exist with particular strengths and weaknesses for different classes of problems. For instance, in the practice of chemical engineering, systems of equations for large-scale separation processes make use of LU factorization to determine aqueous and organic compositions. A variant of LU factorization is the Cholesky factorization; this method requires a matrix, A, to be symmetric and positive definite, such that it can be reduced to a product of a lower triangular matrix and its transpose, $A = LL^T$. Furthermore, this method finds its application in linear least squares regression, matrix inversion, and partial differential equation boundary value problems. [1]

In this investigation, the Cholesky factorization is used to construct the concentration profile for a two-dimensional mass transport problem. Following the rigorous application of mass transport principles, the continuity equation reduces to an elliptic partial differential equation that can be solved using a finite difference approach for internal nodes. Overall, a comparison of the analytical and numerical results of the problem statement are used to facilitate discussion on the advantages, disadvantages, and recommendations for the method.

Theoretical Principles

The reader should note that the present discussion is limited to the analysis of real matrices. The motivation for a need of the matrix decomposition technique is that factoring a matrix will solve the system of equations, which will save computational runtime in the event of solving for various inputs.

There are many different types of matrix decompositions. Analogous to LU decomposition, Cholesky decomposition is a matrix factorization technique which decomposes a real matrix, A, to a product of a lower triangular matrix with its transpose:

$$A = L \cdot L^T$$

This technique may be employed if A is positive-definite and symmetric. [2] There are three conditions to check for if a matrix satisfies the mentioned criteria.

If the matrix is square:

$$[n, m] = size(A)$$

$$if\ n == m$$

If the matrix is symmetric:

$$if\ A == A'$$

If the matrix is positive-definite [3]:

$$if\ \det(A) > 0$$

If the computational criteria are satisfied, the following formulas (derivation found in appendices) may be utilized to determine the matrix elements of *L*:

$$L_{11} = \sqrt{a_{11}}, i = 1$$

$$L_{j1} = \frac{a_{1j}}{L_{11}}, i = 1 \; and \; j \in [2, n]$$

$$L_{ii} = \sqrt{a_{ii} - \sum_{j=1}^{i-1} L_{ij}^2}, i > 1$$

$$L_{ji} = \frac{a_{ij} - \sum_{k=1}^{i-1} L_{ik} \cdot L_{jk}}{L_{ii}}, i > 1 \; and \; j \in (i, n]$$

The last two formulae may be rewritten to utilize *MATLAB*'s matrix operation capabilities:

$$L_{ii} = \sqrt{a_{ii} - L(i, 1{:}\, i-1) \cdot L(i, 1{:}\, i-1)^T}$$

$$L_{ji} = \frac{a_{ij} - L(i, 1{:}\, i-1) \cdot L(j, 1{:}\, i-1)^T}{L_{ii}}$$

Discussion

The problem statement for the mass transport application was as follows:

The device shown is used to continuously release volatile component A into a flowing air stream by a diffusion-limited process. A reservoir that serves as a source for solid A rests on top of a porous slab of dimensions L = 4.0 cm in the x-direction and H = 6.0 cm in the y-direction. At the top of the porous slab, the solid sublimes, providing a vapour at saturation vapour pressure of 0.10 atm and a concentration of C_A*. The vapour diffuses through the porous slab, and then out through the exposed faces of the slab. Assume that volatile species A is immediately dispersed into the air stream maintained at 27 °C and 1.0 atm so that C_{AS} = 0. Determine an equation for the concentration profile of *A* within the porous material. [4]

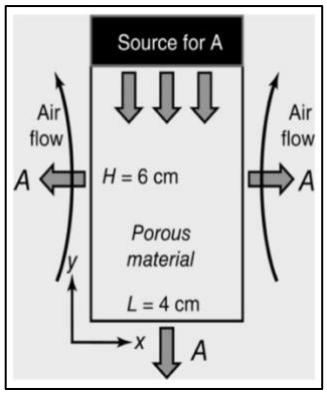


Figure 1: *Problem Statement Diagram*

Analytically, the equation for the concentration profile of *A* was determined per the assumptions and rearrangement leading to the analytical equation in Appendix E. Numerically, following the derivation of the continuity equation and identifying it as an elliptic partial differential equation, central finite difference formulas were applied to internal nodes to obtain the general node equation in Appendix F. Then, Cholesky decomposition was applied. Ultimately, for practicality, construction of the concentration profile was performed via MATLAB for the analytical and numerical results; identical contour plots of the concentration profile were observed per Figure 2.

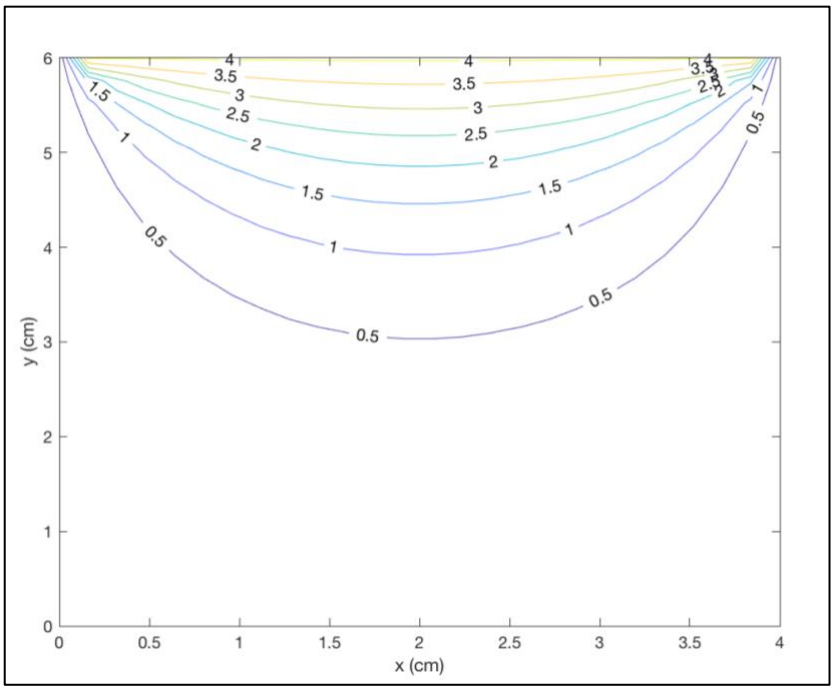


Figure 2: *Contour Plot of the 2D Steady-State Mass Transfer Problem using Finite Differences and Cholesky Decomposition with 576 Internal Nodes, and via the Analytical Solution*

In-depth analytical and numerical solutions were performed as shown in Appendix E and F. From Figure 2, it can be observed that the results of the analytical and numerical solutions were identical; it was noted that using the Cholesky method does not require deriving and solving complex partial differential equations. This validated the accuracy and effectiveness of the Cholesky method when solving similar partial differential equations.

Cholesky decomposition is often compared to LU decomposition due to the shared properties of the two methods. The biggest drawback to Cholesky decomposition is the limitation on the applicable matrices. Symmetric, positive-definite matrices are not common, although they can be used for numerical optimization, linear programming on GPUs, and chemical engineering applications. [5,6] However, matrices need to be checked to be positive-definite and symmetric before using Cholesky method as opposed to being directly inputted into code. In addition, Cholesky decomposition utilizes square roots and squares of elements, which can increase the amount of calculation in comparison to LU decomposition and Gaussian elimination. Just like other numerical methods, the Cholesky method, despite being very accurate, creates truncation error and roundoff error.

Cholesky decomposition is much more efficient compared to LU decomposition if the system in question is symmetric and positive-definite. For the problem statement, it was chosen to solve the system using 25 steps which resulted in 24 nodes across the x-axis and 24 nodes across the y-axis. In total, there were 576 internal nodes. The systems of equations was solved using both Cholesky and LU factorization. The Cholesky method was consistently faster than LU factorization in terms of computational runtime. Like LU factorization, Cholesky decomposition created two matrices during calculation. However, its second matrix was the transpose of the lower triangular matrix; the amount of computations for decomposition was effectively reduced because a matrix and its transpose were determined as opposed to two distinct matrices. Using one less matrix also reduced the storage requirement by half, which would be significant for large matrices.

Like LU decomposition, Cholesky decomposition also stores data from the calculation process and allows for future reference in case similar process were to be used, further saving work to be done in the future. Cholesky method, unlike LU decomposition, generally does not require pivoting, making the matrix much more stable. Finally, as a numerical method, Cholesky decomposition allows for complex calculations to be conducted via matrices, thus bypassing rigorous analytical solutions.

Conclusion

In final analysis, the Cholesky factorization was implemented successfully for a mass transport problem, alongside an analytical solution. Overall, the method excelled in exploiting the symmetry and positive-definite characteristics of a system; computationally, for the problem, the method's runtime was faster than that of LU factorization by 0 to 0.5 seconds on average. The disadvantage was the applicability of the method as it required unique matrix properties. Finally, per Figure 2, the contour plots generated were identical, verifying the successful implementation of analytical and numerical methods. A recommendation for the application of this method would be to ensure that the system of equations, prior to factorization, is symmetric and positive-definite. After verifying that $A = A^T$ for symmetry, A can be reduced to a triangular matrix with its diagonal elements as the eigenvalues; if all eigenvalues are positive, then the system is solvable via Cholesky decomposition. This method should be implemented to achieve minimal runtime and high computational efficiency. Other recommendations, for future studies, include engaging in the rigorous error analysis of Cholesky factorization and comparing it with LU factorization where the system of equations is not sparse.

References

[1] J. Browniee, "A Gentle Introduction to Matrix Factorization for Machine Learning," *Machine Learning Mastery*, 07-Apr-2018. [Online]. Available: https://machinelearningmastery.com/introduction-to-matrix-decompositions-for-machine-learning/. [Accessed: 17-Jul-2018].

[2] Kenneth. Beers, *Numerical Methods for Chemical Engineering – Applications in MATLAB*. Cambridge, United Kingdom: Cambridge University Press, 2006, ch. 1, sec. 7.2, pp.42. Available From: https://app.knovel.com/hotlink/pdf/id:kt008EZQM4/numerical-methods-chemical/cholesky-decomposition

[3] Eric. Weisstein. *Positive Definite Matrix*. Available From: http://mathworld.wolfram.com/PositiveDefiniteMatrix.html

[4] J. Welty, C. E. Wicks, G. L. Rorrer, and R. E. Wilson, Fundamentals of Momentum, Heat and Mass Transfer. s.l.: John Wiley & Sons Ltd, 2013.

[5] Beers, K. J. (2009). *Numerical methods for chemical engineering: Applications in MATLAB*. Cambridge: Cambridge University Press.

[6] Jung, J. H. (n.d.). Cholesky Decomposition and Linear Programming on a GPU. Retrieved July 16, 2018, from https://pdfs.semanticscholar.org/f154/77aaec77a6aee5061f8f9a14a2eedd9f7f7c.pdf

[7] M.A. Al-Gwaiz, *Sturm-Liouville Theory and its Applications*. London, United Kingdom: Springer, 2008, ch. 2, sec. 3, 4, pp.60, 67.

Appendices

A. Regarding the Computation and Implementation of Cholesky Factorization

It is sufficient to use 3 *x* 3 matrices in the investigation of the computation and implementation of Cholesky decomposition:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ L_{31} & L_{32} & L_{33} \end{bmatrix} \cdot \begin{bmatrix} L_{11} & L_{21} & L_{31} \\ 0 & L_{22} & L_{32} \\ 0 & 0 & L_{33} \end{bmatrix}$$

After conducting matrix multiplication of the R.H.S., the following equations will be used to solve for the elements of the factor. The equations will be annotated with the newly determined element or *R* for redundant:

$$a_{11} = L_{11}^2 \colon\ L_{11}$$

$$a_{12} = L_{11}L_{21} \colon\ L_{21}$$

$$a_{13} = L_{11}L_{31} \colon\ L_{31}$$

$$a_{21} = L_{11}L_{21} \colon\ R$$

$$a_{22} = L_{21}^2 + L_{22}^2 \colon\ L_{22}$$

$$a_{23} = L_{21}L_{31} + L_{22}L_{32} \colon\ L_{32}$$

$$a_{31} = L_{31}L_{11} \colon\ R$$

$$a_{32} = L_{31}L_{21} + L_{32}L_{22} \colon\ R$$

$$a_{33} = L_{31}^2 + L_{32}^2 + L_{33}^2 \colon\ L_{33}$$

Since there are redundant equations, there are less computations required than there appears to be superficially. If the factor elements are determined in the order of the first column, down to the n^{th} row, moving onto the next column starting from the diagonal, and repeated to the n^{th} diagonal, the number of computations required for this system is:

$$n_c = 3 + 2 + 1 = 6$$

For a 4 *x* 4 matrix:

$$n_c = 4 + 3 + 2 + 1 = 10$$

And in general:

$$n_c = \sum_{i=1}^{n} i$$

In practice, the above equation is tedious to compute. A much simpler equation may be employed for this task.

For 10 *x* 10 matrices:

$$n_c = 10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1$$

Note that:

$$n_c = 10 + (9 + 1) + (8 + 2) + (7 + 3) + (6 + 4) + 5$$

$$= 10 + 10 + 10 + 10 + 10 + 5 = 5 \cdot 10 + 5 = 55$$

For 11 *x* 11 matrices:

$$n_c = 11 + 10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1$$

$$= 11 + (10 + 1) + (9 + 2) + (8 + 3) + (7 + 4) + (6 + 5)$$

$$= 11 + 11 + 11 + 11 + 11 + 11 = 6 \cdot 11 = 66$$

In general, the formulae for matrix dimension, *n*:

$$n_c = n \cdot \frac{n + 1}{2}$$

For the 2D steady-state mass transfer problem in this discussion, there are 24 increments which were used for a finite difference approach of solving the equation of continuity. Hence the resulting systems of equation is a 576 *x* 576 (24^2) matrix. Then the number of computations required to determine all the factor elements is:

$$n_c = 576 \cdot \frac{576 + 1}{2} = 166176$$

B. Derivation of the Cholesky Element Equations

A 4 *x* 4 will be an adequate enough sized matrix for deriving general formulae:

$$A = \begin{bmatrix} L_{11} & 0 & 0 & 0 \\ L_{21} & L_{22} & 0 & 0 \\ L_{31} & L_{32} & L_{33} & 0 \\ L_{41} & L_{42} & L_{43} & L_{44} \end{bmatrix} \cdot \begin{bmatrix} L_{11} & L_{21} & L_{31} & L_{41} \\ 0 & L_{22} & L_{32} & L_{42} \\ 0 & 0 & L_{33} & L_{43} \\ 0 & 0 & 0 & L_{44} \end{bmatrix}$$

$$= \begin{bmatrix} L_{11}^2 & L_{11}L_{21} & L_{11}L_{31} & L_{11}L_{41} \\ \cdots & L_{21}^2 + L_{22}^2 & L_{21}L_{31} + L_{22}L_{32} & L_{21}L_{41} + L_{22}L_{42} \\ \cdots & \cdots\cdots\cdots\cdots & L_{31}^2 + L_{32}^2 + L_{33}^2 & L_{31}L_{41} + L_{32}L_{42} + L_{33}L_{43} \\ \cdots & \cdots\cdots\cdots\cdots & \cdots & L_{41}^2 + L_{42}^2 + L_{43}^2 + L_{44}^2 \end{bmatrix}$$

Assuming *A* is a real, positive-definite matrix, the four formulae in the discussion could be derived by comparing the coefficients of $L \cdot L^T$ with *A*. The first formula is essentially obtained for free:

$$a_{11} = L_{11}^2$$

$$\therefore L_{11} = \sqrt{a_{11}}$$

The second formula could be obtained by comparison of the top row elements:

$$L_{21} = \frac{a_{12}}{L_{11}}$$

$$L_{31} = \frac{a_{13}}{L_{11}}$$

$$\vdots$$

In general, for a *n x n* matrix:

$$L_{j1} = \frac{a_{1j}}{L_{11}}, j \in [2, n]$$

The range for *j* is set as specified since L_{11} has its own formula. The third formula could be obtained by looking at the i^{th} diagonal where $i \neq 1$:

$$L_{22} = \sqrt{a_{22} - L_{21}^2}$$

$$L_{33} = \sqrt{a_{33} - L_{31}^2 - L_{32}^2}$$

$$\vdots$$

$$\therefore L_{ii} = \sqrt{a_{ii} - \sum_{j=1}^{i-1} L_{ij}^2}, i > 1$$

The last formula may be obtained with slightly more effort:

$$L_{42} = \frac{a_{24} - L_{21}L_{41}}{L_{22}}$$

$$L_{32} = \frac{a_{23} - L_{21}L_{31}}{L_{22}}$$

$$L_{ji} = \frac{a_{ij} - L_{i1}L_{j1}}{L_{ii}} = \frac{a_{ij} - \sum_{k=1}^{1} L_{ik}L_{jk}}{L_{ii}}, for\ i = 2$$

$$L_{43} = \frac{a_{34} - (L_{31}L_{41} + L_{32}L_{42})}{L_{33}} = \frac{a_{34} - \sum_{k=1}^{2} L_{3k}L_{4k}}{L_{33}}$$

$$\therefore\ L_{ji} = \frac{a_{ij} - \sum_{k=1}^{i-1} L_{ik}L_{jk}}{L_{ii}}, (i,j) \in ([2,n],(i,n])$$

The restrictions are put in place to prevent the use of this formula for elements of L in the 1st column and the diagonal elements.

C. Sample Hand Calculation

A positive definite and symmetrical matrix can be solved using Cholesky decomposition as shown below.

$$\begin{bmatrix} 1 & -1 & 2 \\ -1 & 5 & 0 \\ 2 & 0 & 6 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ 3 \\ 1 \end{bmatrix}$$

3 by 3 matrix A is decomposed into the form of $L \cdot L^T$:

$$L_{11} = \sqrt{A_{11}} = 1$$

$$L_{21}L_{11} = A_{21} = -1 \rightarrow L_{21} = -1$$

$${L_{21}}^2 + {L_{22}}^2 = A_{22} = 5 \rightarrow {L_{22}}^2 = 4 \rightarrow L_{22} = 2$$

$$L_{31}L_{11} = A_{31} = 2 \rightarrow L_{31} = 2$$

$$L_{31}L_{21} + L_{32}L_{22} = A_{32} = 0 \rightarrow 2 \times -1 + 2 \times L_{32} = 0 \rightarrow L_{32} = 1$$

$$L_{31}{}^2 + L_{32}{}^2 + L_{33}{}^2 = A_{33} = 6 \rightarrow 2^2 + 1^2 + L_{33}{}^2 = 6 \rightarrow L_{33} = 1$$

$$L = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 2 & 0 \\ 2 & 1 & 1 \end{bmatrix}; L^T = \begin{bmatrix} 1 & -1 & 2 \\ 0 & 2 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Ly = b = \begin{bmatrix} -1 \\ 3 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ -1 & 2 & 0 \\ 2 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} -1 \\ 3 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \\ 2 \end{bmatrix}$$

$$L^T x = y = \begin{bmatrix} -1 \\ 1 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -1 & 2 \\ 0 & 2 & 1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \\ 2 \end{bmatrix} \rightarrow \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -5.5 \\ -0.5 \\ 2 \end{bmatrix}$$

D. Analysis of Chemical Engineering Application

We begin with the equation of continuity:

$$\frac{\partial C_A}{\partial t} + \vec{\nabla} \cdot \overrightarrow{N_A} - R_A = 0$$

We will consider the interior of the porous slab to be our system. The source is the reservoir on top of the slab. The sink is the surrounding air. If the source and sink are assumed to be constant:

$$\vec{\nabla} \cdot \vec{N}_A - R_A = 0$$

In the absence of chemical reactions:

$$\vec{\nabla} \cdot \vec{N}_A = \frac{\partial N_{Ax}}{\partial x} + \frac{\partial N_{Ay}}{\partial y} + \frac{\partial N_{Az}}{\partial z} = 0$$

We will assume the absence of edge effects:

$$\frac{\partial N_{Ax}}{\partial x} + \frac{\partial N_{Ay}}{\partial y} = 0$$

The above equation could be further simplified by using Fick's 1st Law:

$$N_{Ax} = -CD_{AB} \cdot \frac{\partial y_A}{\partial x} + y_A(N_{Ax} + N_{Bx})$$

$$N_{Ay} = -CD_{AB} \cdot \frac{\partial y_A}{\partial y} + y_A(N_{Ay} + N_{By})$$

If diffusion is to occur under isobaric and isothermal conditions:

$$N_{Ax} = -D_{AB} \cdot \frac{\partial C_A}{\partial x} + y_A(N_{Ax} + N_{Bx})$$

$$N_{Ay} = -D_{AB} \cdot \frac{\partial C_A}{\partial y} + y_A(N_{Ay} + N_{By})$$

Species *B* is stagnant:

$$N_{Ax} = -D_{AB} \cdot \frac{\partial C_A}{\partial x} + y_A N_{Ax}$$

$$N_{Ay} = -D_{AB} \cdot \frac{\partial C_A}{\partial y} + y_A N_{Ay}$$

We will neglect advective effects to ensure an analytical solution may be obtained:

$$N_{Ax} = -D_{AB} \cdot \frac{\partial C_A}{\partial x}$$

$$N_{Ay} = -D_{AB} \cdot \frac{\partial C_A}{\partial y}$$

Substituting the simplified Fick's Law relations into the continuity equation:

$$-D_{AB} \cdot \frac{\partial^2 C_A}{\partial x^2} - D_{AB} \cdot \frac{\partial^2 C_A}{\partial y^2} = 0$$

The following equation will be the basis for further calculations (both analytical and numerical):

$$\frac{\partial^2 C_A}{\partial x^2} + \frac{\partial^2 C_A}{\partial y^2} = 0$$

The partial differential equation is accompanied by the following boundary conditions:

$$C_A(x = 0, y) = 0$$

$$C_A(x = L, y) = 0$$

$$C_A(x, y = 0) = 0$$

$$C_A(x, y = H) = \frac{P^*}{RT} = \frac{0.1\ atm}{\frac{16}{195}\frac{atm \cdot L}{mol \cdot K} \cdot (27 + 273.15)\ K} = 4.06 \cdot 10^{-3}\frac{mol}{L}$$

E. Analytical Solution

The analytical solution may be obtained through the technique of Separations of Variables. We begin by assuming the concentration profile may be separated into a function of *x* and as a function of *y*:

$$C_A(x, y) = X(x) \cdot Y(y)$$

The partial derivatives are computed as follows:

$$\frac{\partial C_A}{\partial x} = Y(y) \cdot \frac{dX}{dx}$$

$$\frac{\partial^2 C_A}{\partial x^2} = Y(y) \cdot \frac{d^2 X}{dx^2}$$

$$\frac{\partial C_A}{\partial y} = X(x) \cdot \frac{dY}{dy}$$

$$\frac{\partial^2 C_A}{\partial y^2} = X(x) \cdot \frac{d^2 Y}{dy^2}$$

We substitute the 2nd order partial derivatives into the continuity equation to yield:

$$X \cdot \frac{d^2 Y}{dy^2} + Y \cdot \frac{d^2 X}{dx^2} = 0$$

$$X \cdot \frac{d^2Y}{dy^2} = -Y \cdot \frac{d^2X}{dx^2}$$

$$\frac{1}{X} \cdot \frac{d^2X}{dx^2} = -\frac{1}{Y} \cdot \frac{d^2Y}{dy^2} = \lambda$$

The reader should note that λ represents a constant. The statement above is true as the L.H.S. is a function of *x* and the R.H.S. is a function of *y*. Inequality would imply that the L.H.S. and the R.H.S. are not only functions of *x* and *y*, respectively.

There are three classes of values which λ may take: negative, positive, and identically zero. The reader should note that the authors have checked the case of λ being positive and identically zero and verified that those two cases produce the trivial solution. The trick for choosing the sign of the constant was to identify which coordinate is symmetric with respect to the boundaries (in this case, *x*) and then choose the sign which will produce periodic functions for that coordinate.

Moving on, in the case that λ is negative:

$$\frac{1}{X} \cdot \frac{d^2X}{dx^2} = -a^2$$

$$\frac{d^2X}{dx^2} + a^2X = 0$$

$$X(x) = Acos(ax) + Bsin(bx)$$

$$\frac{-1}{Y} \cdot \frac{d^2Y}{dy^2} = -a^2$$

$$\frac{d^2Y}{dy^2} - a^2Y = 0$$

$$Y(y) = Ccosh(ay) + Bsinh(ay)$$

Our concentration profile takes on the general form of:

$$C_A(x, y) = (Acos(ax) + B(\sin(ax)) \cdot (Ccosh(ay) + Dsinh(ay))$$

We could now apply our boundary conditions to determine the value of the constants:

$$C_A(x = 0, y) = A \cdot \big(Ccosh(ay) + Dsinh(ay)\big) = 0$$

For this relationship to be identically true, $A = 0$, resulting in:

$$C_A(x, y) = \sin(ax) \cdot (B'cosh(ay) + C'sinh(ay))$$

Applying another boundary condition:

$$C_A(x, y = 0) = B' \sin(ax) = 0$$

$$\therefore B' = 0$$

Applying another homogeneous boundary condition:

$$k = C'$$
$$C_A(x = L, y) = k \cdot \sin(aL) \cdot \sinh(ay) = 0$$

To avoid the trivial solution:

$$sin(aL) = 0$$

$$aL = n\pi$$

$$a_n = \frac{n\pi}{L}$$

The concentration profile may be expressed as:

$$C_A(x, y) = \sum_{n=1}^{\infty} k_n \cdot \sin\left(\frac{n\pi}{L}x\right) \cdot \sinh(\frac{n\pi}{L} \cdot y)$$

All that remains is the value of the constants, k_n. Using our last boundary condition:

$$C_A(x, y = H) = \sum_{n=1}^{\infty} k_n \cdot \sin\left(\frac{n\pi}{L}x\right) \cdot \sinh\left(\frac{n\pi H}{L}\right) = C_A^*$$

The orthogonality properties of *sin* functions may be utilized as our concentration profile could be written in terms of Sturm-Liouville form and our boundary conditions satisfies the necessary boundary conditions [7]:

$$\begin{cases} \int_0^L \sin\left(\frac{n\pi}{L}x\right) \cdot \sin\left(\frac{m\pi}{L}x\right) dx = \frac{L}{2}, if\ n = m \\ \int_0^L \sin\left(\frac{n\pi}{L}x\right) \cdot \sin\left(\frac{m\pi}{L}x\right) dx = 0, if\ n \neq m \end{cases}$$

Back to the topic at hand:

$$C_A^* = \sum_{n=1}^{\infty} k_n \cdot \sin\left(\frac{n\pi}{L}x\right) \cdot \sinh\left(\frac{n\pi H}{L}\right)$$

$$\sin(\frac{m\pi}{L}x)\cdot C_A^* = \sin(\frac{m\pi}{L}x)\cdot\sum_{n=1}^{\infty} k_n\cdot\sin\left(\frac{n\pi}{L}x\right)\cdot\sinh\left(\frac{n\pi H}{L}\right)$$

$$\int_0^L \sin\left(\frac{m\pi}{L}x\right)\cdot C_A^*\cdot dx = \int_0^L\sum_{n=1}^{\infty} k_n\cdot\sin(\frac{m\pi}{L}x)\cdot\sin\left(\frac{n\pi}{L}x\right)\cdot\sinh\left(\frac{n\pi H}{L}\right)\cdot dx$$

By the properties of integrals:

$$=\sum_{n=1}^{\infty}\int_0^L k_n\cdot\sin(\frac{m\pi}{L}x)\cdot\sin\left(\frac{n\pi}{L}x\right)\cdot\sinh\left(\frac{n\pi H}{L}\right)\cdot dx$$

$$= k_1\cdot\sinh(\frac{\pi H}{L})\cdot\int_0^L \sin\left(\frac{m\pi}{L}x\right)\cdot\sin\left(\frac{\pi}{L}x\right)\cdot dx + k_2\cdot\sinh\left(\frac{2\pi H}{L}\right)\cdot\int_0^L \sin\left(\frac{m\pi}{L}x\right)\cdot\sin\left(\frac{2\pi}{L}x\right)\cdot dx + \ldots + k_m\cdot\sinh(\frac{m\pi H}{L})\cdot\int_0^L \sin\left(\frac{m\pi}{L}x\right)^2 dx + \cdots$$

$$= 0+0+\cdots+k_m\cdot\sinh\left(\frac{m\pi H}{L}\right)\cdot\frac{L}{2}+0+0+\cdots$$

$$k_m\cdot\frac{L}{2}\cdot\sinh\left(\frac{m\pi H}{L}\right) = -C_A^*\cdot\frac{L}{m\pi}\cdot\cos(\frac{m\pi}{L}x)\Big|\begin{matrix}x=L\\x=0\end{matrix}$$

$$= -C_A^*\cdot\frac{L}{m\pi}\cdot((-1)^m-1)$$

$$k_m = \frac{2C_A^*\cdot(1-(-1)^m)}{m\pi\cdot\sinh(\frac{m\pi H}{L})}$$

Therefore, our final expression for the concentration profile of species *A* within the porous slab is:

$$C_A(x,y) = 2C_A^*\cdot\sum_{n=1}^{\infty}\frac{(1-(-1)^n)}{n\pi\cdot\sinh(\frac{n\pi H}{L})}\cdot\sin\left(\frac{n\pi}{L}x\right)\cdot\sinh(\frac{n\pi}{L}y)$$

F. Numerical Solution

We begin with the simplified equation of continuity:

$$\frac{\partial^2 C_A}{\partial x^2} + \frac{\partial^2 C_A}{\partial y^2} = 0$$

Instead of Separation of Variables, however, we may use a Finite Difference solution which solves an elliptic PDE with central difference formulas at internal "nodes". Let *i* and *j* represent the *x* and *y* coordinates, respectively:

$$\frac{\partial^2 C_A}{\partial x^2} = \frac{C_{A_{i+1,j}} - 2C_{A_{i,j}} + C_{A_{i-1,j}}}{\Delta x^2}$$

$$\frac{\partial^2 C_A}{\partial y^2} = \frac{C_{A_{i,j+1}} - 2C_{A_{i,j}} + C_{A_{i,j-1}}}{\Delta y^2}$$

Substituting the central difference formulas into the equation of continuity:

$$\frac{C_{A_{i+1,j}} - 2C_{A_{i,j}} + C_{A_{i-1,j}}}{\Delta x^2} + \frac{C_{A_{i,j+1}} - 2C_{A_{i,j}} + C_{A_{i,j-1}}}{\Delta y^2} = 0$$

The reader should note that the simplification of using equal Δx and Δy cannot be made due to the length of the slab not being identical in the *x* and *y* direction if a *n x n* matrix is to be constructed. Returning to the topic at hand, the general node equation is:

$$-\Delta x^2 \cdot C_{A_{i,j-1}} - \Delta y^2 \cdot C_{A_{i-1,j}} + 2(\Delta x^2 + \Delta y^2) \cdot C_{A_{i,j}} - \Delta y^2 \cdot C_{A_{i+1,j}} - \Delta x^2 \cdot C_{A_{i,j+1}} = 0$$

The negative of the equation is taken so that the resulting matrix is positive-definite. There are a few special nodes (where boundary values apply) which could be taken care of.

For the corner nodes:

N (1 , 1)

$$2(\Delta x^2 + \Delta y^2) \cdot C_{A_{i,j}} - \Delta y^2 \cdot C_{A_{i+1,j}} - \Delta x^2 \cdot C_{A_{i,j+1}} = 0$$

N (n , 1)

$$-\Delta y^2 \cdot C_{A_{i-1,j}} + 2(\Delta x^2 + \Delta y^2) \cdot C_{A_{i,j}} - \Delta x^2 \cdot C_{A_{i,j+1}} = 0$$

N (1 , n)

$$-\Delta x^2 \cdot C_{A_{i,j-1}} + 2(\Delta x^2 + \Delta y^2) \cdot C_{A_{i,j}} - \Delta y^2 \cdot C_{A_{i+1,j}} = \Delta x^2 \cdot \frac{P^*}{RT}$$

N (n , n)

$$-\Delta x^2 \cdot C_{A_{i,j-1}} - \Delta y^2 \cdot C_{A_{i-1,j}} + 2(\Delta x^2 + \Delta y^2) \cdot C_{A_{i,j}} = \Delta x^2 \cdot \frac{P^*}{RT}$$

For the 1st layer which excludes the corners:

N (2 – (n-1) , 1)

$$-\Delta y^2 \cdot C_{A_{i-1,j}} + 2(\Delta x^2 + \Delta y^2) \cdot C_{A_{i,j}} - \Delta y^2 \cdot C_{A_{i+1,j}} - \Delta x^2 \cdot C_{A_{i,j+1}} = 0$$

N (1 , 2 – (n-1))

$$-\Delta x^2 \cdot C_{A_{i,j-1}} + 2(\Delta x^2 + \Delta y^2) \cdot C_{A_{i,j}} - \Delta y^2 \cdot C_{A_{i+1,j}} - \Delta x^2 \cdot C_{A_{i,j+1}} = 0$$

N (2 – (n-1) , n)

$$-\Delta x^2 \cdot C_{A_{i,j-1}} - \Delta y^2 \cdot C_{A_{i-1,j}} + 2(\Delta x^2 + \Delta y^2) \cdot C_{A_{i,j}} - \Delta y^2 \cdot C_{A_{i+1,j}} = \Delta x^2 \cdot \frac{P^*}{RT}$$

N (n , 2 – (n-1))

$$-\Delta x^2 \cdot C_{A_{i,j-1}} - \Delta y^2 \cdot C_{A_{i-1,j}} + 2(\Delta x^2 + \Delta y^2) \cdot C_{A_{i,j}} - \Delta x^2 \cdot C_{A_{i,j+1}} = 0$$

For example, a solution with 4 internal nodes will only require the corner equations, but it will be sufficient for demonstration purposes. In this situation:

$$\Delta x = \frac{L}{2+1} = \frac{4}{3}\ cm$$

$$\Delta y = \frac{H}{2+1} = \frac{6}{3} = 2\ cm$$

Our systems of equations will be (in order of (1,1), (2,1), (1,2), and (2,2)):

$$\frac{104}{9} C_{A_{1,1}} - 4C_{A_{2,1}} - \frac{16}{9} C_{A_{1,2}} = 0$$

$$-4C_{A_{1,1}} + \frac{104}{9} C_{A_{2,1}} - \frac{16}{9} C_{A_{2,2}} = 0$$

$$-\frac{16}{9} C_{A_{1,1}} + \frac{104}{9} C_{A_{1,2}} - 4C_{A_{2,2}} = \frac{16}{9} \cdot \frac{0.1}{\frac{16}{195} \cdot 300.15} \cdot \frac{1}{1000}$$

$$-\frac{16}{9} C_{A_{2,1}} - 4C_{A_{1,2}} + \frac{104}{9} C_{A_{2,2}} = \frac{16}{9} \cdot \frac{0.1}{\frac{16}{195} \cdot 300.15} \cdot \frac{1}{1000}$$

In a matrix:

$$\begin{bmatrix} \frac{104}{9} & -4 & -\frac{16}{9} & 0 \\ -4 & \frac{104}{9} & 0 & -\frac{16}{9} \\ -\frac{16}{9} & 0 & \frac{104}{9} & -4 \\ 0 & -\frac{16}{9} & -4 & \frac{104}{9} \end{bmatrix} \cdot \begin{bmatrix} C_{A_{1,1}} \\ C_{A_{2,1}} \\ C_{A_{1,2}} \\ C_{A_{2,2}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 7.218 \cdot 10^{-6} \\ 7.218 \cdot 10^{-6} \end{bmatrix}$$

Our matrix should be checked for symmetry if Cholesky Factorization is to be used:

$$A - A^T = 0$$

Positive-definite matrices have positive determinants. Using *MATLAB*:

$$\det(A) = 1.2878 \cdot 10^4$$

We could now perform Cholesky Factorization:

$$L_{1,1} = \sqrt{a_{11}} = \sqrt{\frac{104}{9}} = 3.399$$

$$L_{2,1} = -\frac{4}{3.399} = -1.177$$

$$L_{3,1} = \frac{-\frac{16}{9}}{3.399} = -0.5230$$

$$L_{4,1} = \frac{0}{3.399} = 0$$

$$L_{2,2} = \sqrt{\frac{104}{9} - (-1.177)^2} = 3.189$$

$$L_{3,2} = \frac{0 - (-1.177) \cdot (-0.5230)}{3.189} = -0.1930$$

$$L_{4,2} = \frac{-\frac{16}{9} - (-1.177) \cdot (0)}{3.189} = -0.5575$$

$$L_{3,3} = \sqrt{\frac{104}{9} - ((-0.1930)^2 + (-0.5230)^2)} = 3.353$$

$$L_{4,3} = \frac{-4 - ((0) \cdot (-0.5230) + (-0.5575) \cdot (-0.1930))}{3.353} = -1.225$$

$$L_{4,4} = \sqrt{\frac{104}{9} - ((0)^2 + (-0.5575)^2 + (-1.225)^2)} = 3.122$$

The Cholesky lower triangular matrix is:

$$L = \begin{bmatrix} 3.399 & 0 & 0.000 & 0.000 \\ -1.117 & 3.189 & 0.000 & 0.000 \\ -0.5230 & -0.1930 & 3.353 & 0 \\ 0 & -0.5575 & -1.225 & 3.122 \end{bmatrix}$$

Our original systems of equations may be represented as:

$$A \cdot \overrightarrow{C_A} = (L \cdot L^T) \cdot \overrightarrow{C_A}$$

Then the concentration at the internal nodes is determined as follows:

$$L^{-1} \cdot (L \cdot L^T) \cdot \overrightarrow{C_A} = L^{-1} \cdot \vec{b}$$

$$L^{-1} \cdot \vec{b} = \vec{d}$$

$$L^T \cdot \overrightarrow{C_A} = \vec{d}$$

$$(L^T)^{-1} \cdot L^T \cdot \overrightarrow{C_A} = (L^T)^{-1} \cdot \vec{d}$$

$$\overrightarrow{C_A} = (L^T)^{-1} \cdot (L)^{-1} \cdot \vec{b}$$

G. Code for Cholesky Factorization

```
function L = cholesky(A)
```

```
  [n,m] = size(A);

  if n ~= m
      error('abort! Matrix not square')
  end

  if A-1/2*(A+A') ~= zeros(n,n)
      error('matrix not symmetric')
  end

  if det(A) < 0
      error('matrix is not positive-definite')
  end

  L = zeros(n,n);

  for i = 1:n     % column
     for j = i:n % row
         % computing the diagonals
         if j == i
             if j == 1
                 L(j,i) = sqrt(A(j,i));
             else
                 L(j,i) = sqrt(A(j,i) - L(i,1:i-1)*L(i,1:i-1)');
             end
         % elements left of diagonals
         else % j < i
             if i == 1
                L(j,i) = A(j,i)/L(i,i);
             else
                 L(j,i) = (A(i,j)-L(i,1:i-1)*L(j,1:i-1)')/L(i,i);
             end
         end
     end
  end
end
```

H. Code for Numerical Solution, 2D Steady State Mass Transfer

```
function exploratoryprob
    P = 0.1; % atm
    R = 16/195; % atm*L/mol/K
    T = 27 + 273.15; % K
    Cas = P/(R*T)/1000; % mol/cm^3

    hi = 4/25;
    hj = 6/25;
    xn = 0 + hi:hi:4 - hi; % internal nodes
    n = length(xn); % # of int nodes in across row/column

    A = zeros(n^2,n^2);
    b = zeros(n^2,1);
```

```
for i = 1:n % x
    for j = 1:n % y
        r = n*(j-1)+i; % row/eqn
        c = r; % column

        % dealing with corner nodes
        if (i == 1 || i == n) && (j == 1 || j == n)
            % (1,1) or (n,n)
            if i == j
                % (1,1)
                if i == 1
                    A(r,c) = -2*(hi^2+hj^2);
                    A(r,c+1) = hj^2;
                    A(r,c+n) = hi^2;
                % (n,n)
                else
                    A(r,c-n) = hi^2;
                    A(r,c-1) = hj^2;
                    A(r,c) = -2*(hi^2+hj^2);
                    b(r,1) = -hi^2*Cas;
                end
            % (1,n) or (n,1)
            else
                % (n,1)
                if i == n
                    A(r,c-1) = hj^2;
                    A(r,c) = -2*(hi^2+hj^2);
                    A(r,c+n) = hi^2;
                % (1,n)
                else
                    A(r,c-n) = hi^2;
                    A(r,c) = -2*(hi^2+hj^2);
                    A(r,c+1) = hj^2;
                    b(r,1) = -hi^2*Cas;
                end
            end
        % nodes (1, 2-(n-1))
        elseif i == 1
            A(r,c-n) = hi^2;
            A(r,c) = -2*(hi^2+hj^2);
            A(r,c+1) = hj^2;
            A(r,c+n) = hi^2;
        % nodes (n, 2-(n-1))
        elseif i == n
            A(r,c-n) = hi^2;
            A(r,c-1) = hj^2;
            A(r,c) = -2*(hi^2+hj^2);
            A(r,c+n) = hi^2;
        % nodes (2-(n-1), 1)
        elseif j == 1
            A(r,c-1) = hj^2;
            A(r,c) = -2*(hi^2+hj^2);
            A(r,c+1) = hj^2;
            A(r,c+n) = hi^2;
        % nodes (2-(n-1), n)
        elseif j == n
            A(r,c-n) = hi^2;
```

```
                A(r,c-1) = hj^2;
                A(r,c) = -2*(hi^2+hj^2);
                A(r,c+1) = hj^2;
                b(r,1) = -hi^2*Cas;
            % every other interior node
            else
                A(r,c-n) = hi^2;
                A(r,c-1) = hj^2;
                A(r,c) = -2*(hi^2+hj^2);
                A(r,c+1) = hj^2;
                A(r,c+n) = hi^2;
            end
        end
    end

    A=-A; b=-b;

    t = cputime;
    L = cholesky(A);
    e = cputime-t;
    d = L\b;
    ca = L'\d*100^3; % mol/m^3

    Ca = zeros(length(0:hi:4),length(0:hj:6));
    Ca(2:length(0:hi:4)-1,length(0:hi:4)) = Cas*100^3;

    for i = 1:n
        Ca(2:length(0:hi:4)-1,1+i) = ca((i-1)*n+1:i*n,1);
    end

    [c,h] = contour((0:hi:4)', (0:hj:6)', Ca');
    clabel(c,h)
    xlabel('x (cm)'), ylabel('y (cm)')
    tit = sprintf('Contour Plot (Cholesky %d Nodes - %.2f Seconds).png', n,
e);
    saveas(gcf,tit)

    e1 = cputime-e;
    [L, U] = lu(A);
    e2 = cputime-e1;

    Ca = zeros(length(0:hi:4),length(0:hj:6));
    Ca(2:length(0:hi:4)-1,length(0:hi:4)) = Cas*100^3;
    d = L\b;
    ca_lu = U\d*100^3;

    for i = 1:n
        Ca(2:length(0:hi:4)-1,1+i) = ca_lu((i-1)*n+1:i*n,1);
    end

    [c,h] = contour((0:hi:4)', (0:hj:6)', Ca');
    clabel(c,h)
    xlabel('x (cm)'), ylabel('y (cm)')
    tit = sprintf('Contour Plot (LU %d Nodes - %.2f Seconds).png', n, e2);
    saveas(gcf,tit)
```

```
end
```

I. Code for Analytical Solution

```
function exploratoryAnalyticalSol
global H L cao
H = 6; % Height in cm
L = 4; % Length in cm
cao = 0.1/(8.206*10^(-5)*300.15); % Concentration in mol/m^3


h = H / 25; % Increment in height
l = L / 25; % Increment in length

c = zeros(26,26); % Initializing concentration matrix

% Create concentration matrix
for i = 1:26
   for j = 1:26
       c(i,j) = findsol(l*(i-1),h*(j-1)); % Substitute each x and y into
function
   end
end

% Oops I got the height and length on the opposite sides
c = transpose(c);

% Plot as contour plot
    figure
    [C,h] = contour((0:l:L), (0:h:H), c);
    clabel(C,h)
    xlabel('x (cm)'), ylabel('y (cm)')
end

function s = findsol (x,y)

global H L cao

i = 0;
n = 1;

% Calculating sum part of the function
while n < 99
i = i+((-1)^n-1)/(n*pi*sinh(n*pi*H/L))*sin(n*pi*x/L)*sinh(n*pi*y/L);
n = n + 1;
end

% Multiply by constants before the sum part
s = -2*i*cao;

end
```