Solving the Poisson-Boltzmann Equation with Mimetic Differences

Kim Ngan Huynh and Miguel A. Dumett

April 20, 2024

Publication Number: CSRCR2024-03

Computational Science & Engineering Faculty and Students Research Articles

Database Powered by the Computational Science Research Center Computing Group & Visualization Lab

COMPUTATIONAL SCIENCE & ENGINEERING

SAN DIEGO STATE UNIVERSITY

Computational Science Research Center
College of Sciences
5500 Campanile Drive
San Diego, CA 92182-1245
(619) 594-3430

Solving the Poisson-Boltzmann Equation With Mimetic Differences

Kim Ngan Huynh* and Miguel A. Dumett †‡

April 20, 2024

Abstract

In this paper, the Poisson-Boltzmann equation is solved utilizing mimetic differences and the Mimetic Operator Enhanced Library (MOLE). Several molecules (H_2O, HCl, SO_3, HCN) are investigated.

1 Introduction

The nonlinear Poisson-Boltzmann equation (PBE) is a fundamental equation that is widely used to describe the distribution of electronic potential $u(r)$ in a fluid. PBE is derived from the classical Poisson equation from electrostatics and the Boltzmann distribution from statistical mechanics. The Poisson equation relates the electrostatic potential to charge distribution in a region, and the Boltzmann distribution describes how charged particles distribute in a system that is influenced by an electric potential [2]. The goal of this project is to utilize the MOLE library [1] to numerically estimate the solution of a nonlinear Poisson-Boltzmann equation in 2D using mimetic differences.

2 PDE Description

2.1 2D nonlinear Poisson-Boltzmann Equation

The combination of Poisson equation and Boltzmann distribution allows PBE to model how ionic molecule influences the electrostatic potential in a solvent, crucial for predicting the behavior of ions around charged molecules. There are two regions, Ω_m contains particular ions of the molecule (charges are fixed) and Ω_s contains solvent with dielectric constants (charges are mobile under Boltzmann distribution). The molecular region Ω_m is particularly defined as a circle with a radius of 1 and the solvent region Ω_s as a square with length of 4. The total charge density in the molecule is the summation of all N_m point charges $q_i = z_i e_c$ located at the specific positions r_i, represented by the Dirac delta function $\delta(r)$ in space (Figure 1).

$$-\nabla \cdot (\varepsilon(r)\nabla u(r)) + \bar{\kappa}^2(r)\sinh(u(r)) = \left(\frac{4\pi e_c^2}{k_B T}\right)\sum_{i=1}^{N_m} z_i \delta(r - r_i) \quad \text{in } \Omega \subset \mathbb{R}^2, \tag{1}$$

where

$\varepsilon(r)$ is dimensionless dielectric constant function
$\bar{\kappa}(r)$ is modified Debye-Huckle parameter (modified to be dielectric independent)
e_c is the charge of electron constant
k_B is Boltzmann's constant
T is absolute temperature
z_i are fractions of unit charge

*Computational Science PhD Program at San Diego State University (lhuynh2785@sdsu.edu).
†Editor: Jose E. Castillo
‡Computational Science Research Center at San Diego State University (mdumett@sdsu.edu).

2.2 Domain and Boundary Conditions

Domain:

$$\Omega_s = [-2, 2] \times [-2, 2]$$
$$\Omega_m = \{(x, y);\ x^2 + y^2 \leq 1\}$$

Neumann boundary conditions: $\frac{\partial u}{\partial n} = 0$ on $\partial\Omega$.

Neumann boundary condition specifies that the normal derivative of the potential at the boundary is set to zero. There is no net electric field crossing the boundary. This condition is appropriate for a system of fixed amount of charge as it assumes the electric potential at the boundary does not contribute or interact with the external environment.

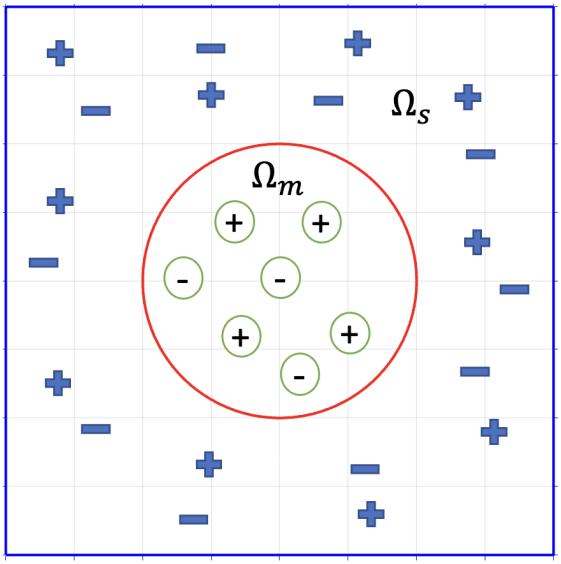

Figure 1: Molecule-solvent system

2.3 Parameter values

$$\varepsilon(r) = \begin{cases} \epsilon_m = 2 & \text{if } r \in \Omega_m, \\ \epsilon_s = 80 & \text{if } r \in \Omega_s. \end{cases}$$

$$\bar{\kappa}(r) = \begin{cases} \bar{\kappa}_m = 0 & \text{if } r \in \Omega_m, \\ \bar{\kappa}_s = \sqrt{\epsilon_s}\kappa & \text{if } r \in \Omega_s. \end{cases}$$

$$\begin{aligned} e_c &= 4.8032424 \times 10^{-10} \text{ esu} \\ k_B &= 1.3806620 \times 10^{-16} \text{ erg/K} \\ T &= 298 \text{ K} \\ \kappa &= 8.486902807 \times \mathring{A}^{-2} \frac{I_s}{\epsilon_s} \\ I_s &= \text{typical ionic strength of 0.1 molar} \end{aligned}$$

2.4 Molecule and Partial Charges

The molecule in Ω_m is chosen to be water which serves as initial conditions. The chemical formula for water is H_2O, which means this molecule has 3 atoms: 2 of hydrogen (H) and 1 oxygen (O) atom. The molecular geometry of water is bent. Since the purpose of this project is to solve nonlinear Poisson-Boltzmann equation in two dimensions, the molecule inside the molecular region Ω_m needs to be a planar molecule. IQmol, a free open source molecular editor and visualization package, is used to symmetrize water to become a planar molecule. The two dimensional positions of three atoms are also collected from IQmol shown in Figure 2. In addition, Q-Chem, an ab initio quantum chemistry program, is employed to calculate the electrostatic potential net atomic charges of water molecule (Figure 3). These atomic charges are fractions of unit charge z_i of (1). Figure 4 illustrates the problem needed to solve for this project. The electrostatic potential of water molecule located inside the molecular region Ω_m within a solvent environment Ω_s containing mobile ions is determined by solving PBE numerically with mimetic differences.

```
$molecule
0 1
  H   -0.0000000    0.4958050    0.7493682
  O   -0.0000000   -0.0156100    0.0000000
  H    0.0000000    0.4958050   -0.7493682
$end
```

Figure 2: Water molecule's coordinates from IQmol

```
   Merz-Kollman ESP Net Atomic Charges

   Atom                 Charge (a.u.)
-----------------------------------------
     1 H                     0.408857
     2 O                    -0.817715
     3 H                     0.408857
-----------------------------------------
 Sum of atomic charges =     0.000000
```

Figure 3: Atomic charges of water molecule in 2D from Q-Chem output file

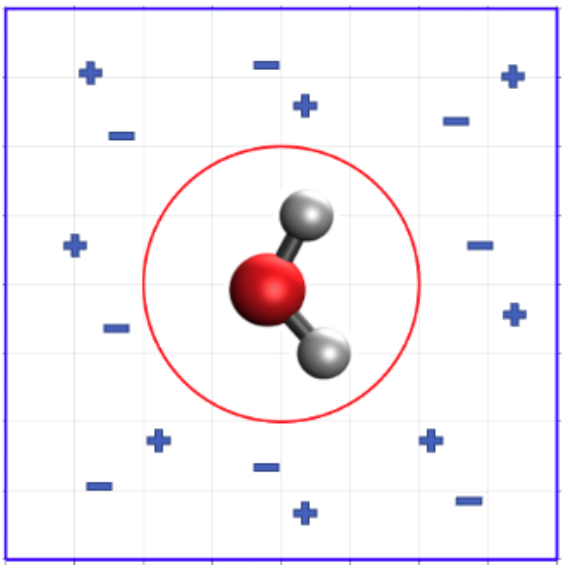

Figure 4: Water molecule-solvent system

3 Mimetic Differences of PBE

For the Poisson-Bolztmann equation

$$-\nabla \cdot (\varepsilon(r)\nabla u(r)) + \bar{\kappa}^2(r)\sinh(u(r)) = \left(\frac{4\pi e_c^2}{k_B T}\right)\sum_{i=1}^{N_m} z_i \delta(r - r_i)$$

the mimetic analog is being constructed step by step. For, U stands for the discrete version of u, already converted to a vector to be able to perform the products with the matrix representation of the mimetic analogs.

After proceeding with the following steps one can attain the discrete analog of equation (1)

1. Replace differential operators:

$$-D(\varepsilon(r)\cdot(GU)) + \bar{\kappa}^2(r)\sinh(U) = \left(\frac{4\pi e_c^2}{k_B T}\right) z_i \delta(r - r_i)$$

2. Match output domains:

$$-D(\varepsilon\cdot(GU)) + \bar{\kappa}^2\sinh(U) = \rho \text{ where } \rho = \left(\frac{4\pi e_c^2}{k_B T}\right) z_i \delta$$

3. Element-wise products:

$$-D(\mathrm{diag}(\varepsilon)\cdot(GU)) + \mathrm{diag}(\bar{\kappa}^2)\sinh(U) = \rho$$

The Poisson-Boltzmann equation in mimetic differences becomes:

$$-D(E\cdot(GU)) + K\sinh(U) = \rho \tag{2}$$

The Dirac delta function on the right-hand-side (RHS) of (1) is not a function, but rather a distribution. The idea of this function is all the values are zero except at a specific given point. In this case, there is a value at each atom located on the staggered grid. In order to satisfy the RHS, radial basis function (RBF) is employed for spatial data interpolation as it computes the similarity of two points.

A Gaussian RBF is typically expressed as

$$f(\mathbf{r}) = e^{-\left(\frac{\|\mathbf{r}-\mathbf{r}_0\|}{\sigma}\right)^2}$$

where $\|\mathbf{r} - \mathbf{r}_0\|$ is the Euclidean distant between point r and the center r_0, and σ is standard deviation of the Gaussian, and can be tuned accordingly.

The divergence (D) and gradient (G) can be generated using `div2D` function and `grad2D` function from the MATLAB version of the MOLE library. The output of these two functions are a matrix $D = \begin{pmatrix} D_x \\ D_y \end{pmatrix}$ and a matrix $G = \begin{pmatrix} G_x \\ G_y \end{pmatrix}$. The constant functions E and K are created using two functions `PartitionOfUnity2Dvector` and `PartitionOfUnity2Dscalar`, respectively. Partition of unity function creates a matrix that contains a value $\epsilon_s = 80$ in most entries. This value will decrease slightly as it gets closer to the molecular region Ω_m. The entries inside the circle are around 2. The output of `PartitionOfUnity2Dvector` is a matrix $\varepsilon = \begin{pmatrix} \varepsilon_h \\ \varepsilon_v \end{pmatrix}$ and `PartitionOfUnity2Dscalar` is a $(m+2)$ by $(n+2)$ matrix $\bar{\kappa}$.

In mimetic difference equation, $-D(E \cdot (GU))$ is a linear part and $K \sinh(U)$ is a nonlinear part. Using the first term in the series expansion $\sinh(x) = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \cdots$ as an approximation of $\sinh(U)$, equation 2 becomes

$$\begin{aligned} (-D(EG) + K) * U &= \rho \\ A * U &= \rho \\ \Rightarrow U &= A \backslash \rho \end{aligned}$$

Applying Newton's method to solve for U the system of nonlinear equations

$$f(U) = -D(E(GU)) + K \sinh(U) - \rho = 0,$$

with an initial guess U from previous step.

Algorithm Mimetic operator pseudocode

1: k = order of accuracy
2: m = number of grid cells in x-axis
3: dx = step size for x
4: n = number of grid cells in y-axis
5: dy = step size for y
6: $[X, Y]$ = 2D Staggered grid
7: x, y = known x,y-position of H_2O
8: $z_i =$ partial charges of H_2O
9: $\delta(r)$ = Radial basis function (Gaussian RBF)
10: ρ = RHS
11: $[\varepsilon_h, \varepsilon_v]$ = **PartitionOfUnity2Dvector()**
12: $E = \text{diag}(\varepsilon_h, \varepsilon_v)$
13: $\bar{\kappa}^2$ = **PartitionOfUnity2Dscalar()**
14: $K = \text{diag}(\bar{\kappa}^2)$
15: D = **div2D()**
16: G = **grad2D()**
17: $A = -D(EG) + K +$ **robinBC2D()**
18: $U_{\text{initial}} = \text{A} \setminus \rho$
19: $f(U) = -D(E(GU)) + Ksinh(U) - \rho$ = Newton's method using $\text{U}_{\text{initial}}$
20: $U = \text{reshape}(U, m+2, n+2)$

4 Results

4.1 Water Molecule

The known points for each atom of water generated from IQmol (Figure 2) and electrostatic potential net atomic charges generated from Q-Chem (Figure 3) are the initial conditions for the Poisson-Boltzmann algorithm. Most articles about solving nonlinear (1) have shown their roofs rather than their numerical solution plots. It is difficult to know what the solution plot of planar water molecule should look like. [2] only shows one solution figure but with a much larger superoxide dismutase (SOD) molecule. Figure 5 (left panel) is the solution plot of planar water molecule obtained initially. However, a significant amount of electrostatic potential is only observed at the position of Hydrogen atom, but not at the position of Oxygen atom. This becomes evident that this numerical solution is not correct.

The position along x-axis and y-axis of each atom, as provided by IQmol is observed to not coincide with the grid points. Upon adjusting these positions to align with the grid points, a distinct peak corresponding to the Oxygen atom becomes evident shown in Figure 5 (right panel). The observation of two positive single peaks at both Hydrogen atom locations and a negative single peak at the location of the Oxygen atom is consistent with the understanding of their respective charge distribution. The Hydrogen atoms have partial positive charges while the Oxygen atom has a negative partial charge.

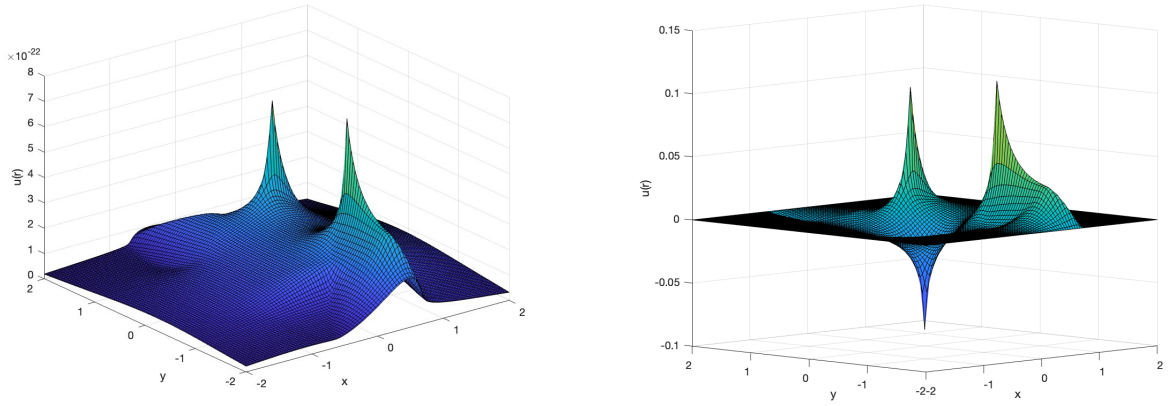

Figure 5: Distribution of electrostatic potential around pre-shifted water molecule (*left*) and shifted water molecule (*right*)

4.2 Other Molecules

As testing the Poisson-Boltzmann algorithm with different initial conditions, three different molecules: HCl, SO_3, HCN are used. Due to the distance between two atoms in these molecules greater than 1, the solvent region is expanded to length of 8 and the radius of molecular region is increased to 2. All atoms of these molecules are shifted to be on the grid points. The correlation between the sign of the partial charge on an atom and the corresponding peak has shown in the solution plots below. A negative peak aligns with the position of an atom carrying a negative partial charge, and a positive peak aligns with the position of an atom carrying a positive partial charge.

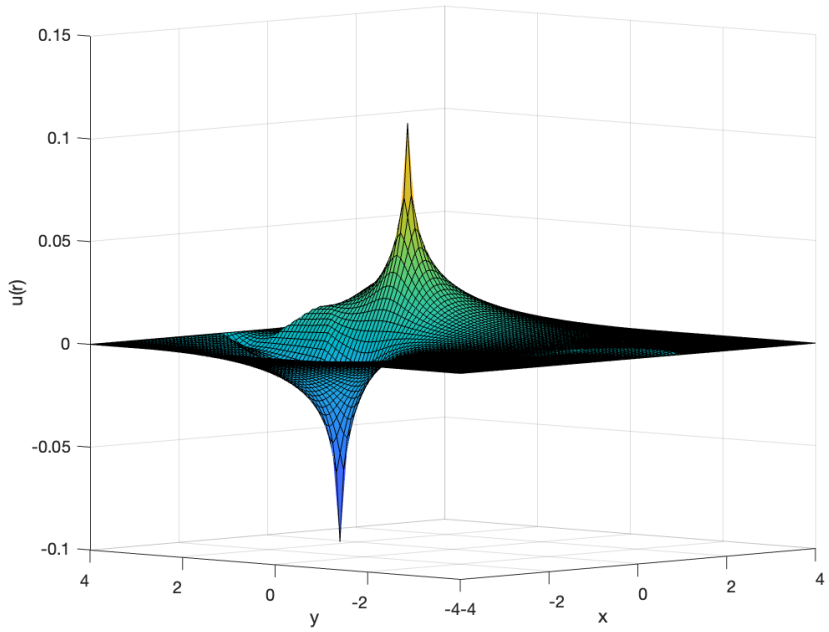

Figure 6: Distribution of electrostatic potential around HCl

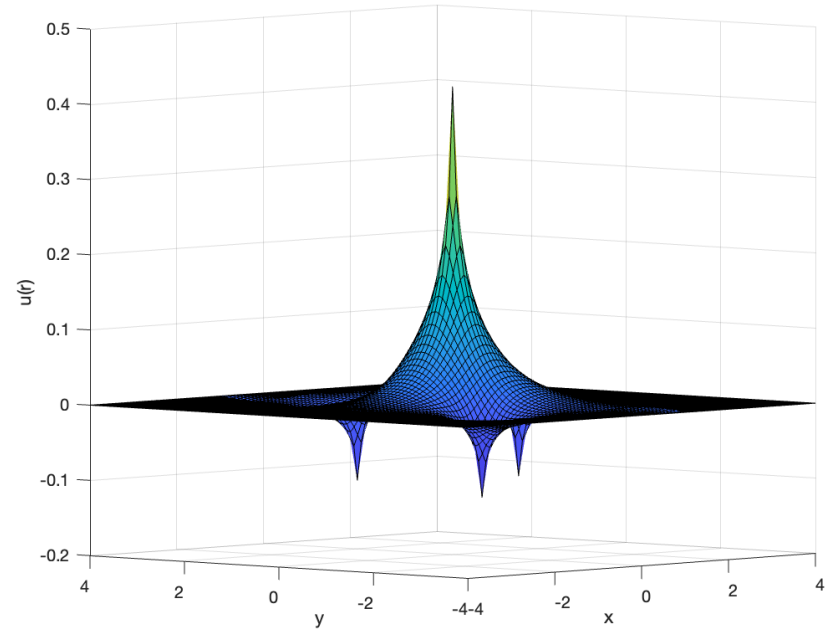

Figure 7: Distribution of electrostatic potential around SO_3

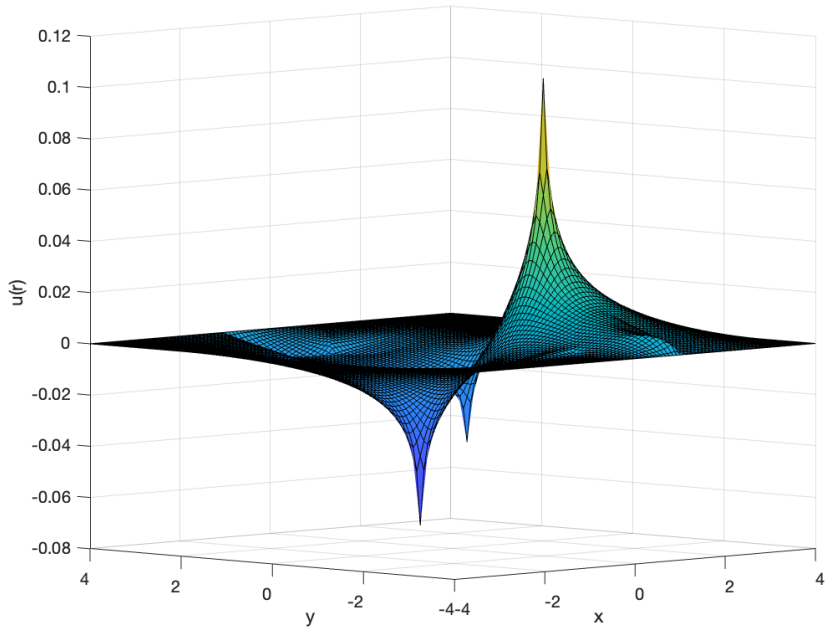

Figure 8: Distribution of electrostatic potential around HCN

Acknowledgements

I would like to thank Professor Yuezhi Mao for his knowledgeable advice on the chemistry aspect.

References

[1] J. Corbino and J. E. Castillo. "MOLE: Mimetic Operators Library Enhanced: The Open-Source Library for Solving Partial Differential Equations using Mimetic Methods". In: (2017). URL: https://github.com/csrc-sdsu/mole.

[2] M.J. Holst. *The Poisson-Boltzmann Equation Analysis and Multilevel Numerical Solution*. 1994. Chap. 1, pp. 1–14.

Appendix

This is input file of a planar water molecule (the positions of each atom are shifted to be on the grid points)

```
$molecule
0 1
  H    0.5000000    0.7400000    0.0000000
  O    0.0200000    0.0200000    0.0000000
  H    0.5000000   -0.7400000    0.0000000
$end

$rem
   BASIS  =  6-31g(D)
   ESP_CHARGES  =  TRUE
   GUI  =  2
   METHOD  =  B3LYP
   SCF_CONVERGENCE  =  8
$end
```

After submitting input file of water molecule to Q-Chem, Merz-Kollman electrostatic potential net atomic charges is obtained as the following

```
Merz-Kollman ESP Net Atomic Charges

    Atom                  Charge (a.u.)
----------------------------------------
    1 H                     0.426725
    2 O                    -0.846350
    3 H                     0.419625
----------------------------------------
Sum of atomic charges =     0.000000
```

The numerical solution of the Poisson-Boltzmann for a planar water molecule shown in Figure 5 (right) was obtained from the MATLAB code below

```
%Solving 2D Poisson-Boltmann Equation with mimetic operators
%PBE in Mimetic differences-D(epsilon(GU)) + (kappa)^2 sinh(U) = rho
addpath '/Users/lunahuynh/SDSU-23-24/COMP670/mole/mole_MATLAB'

%Parameters
k=2;            %order of accuracy
m=100;          %number of cells in x-axis
n=m;            %number of cells in y-axis
a=-2;           %west
b=2;            %east
c=-2;           %south
d=2;            %north
dx=(b-a)/m;
dy=(d-c)/n;

I=0.1;          %initial condition for ionic strength
C=7046.528885; %coefficient on the RHS of PBE

%Dielectric constant function epsilon
epsilon_s = 80;
epsilon_m = 2;

%Debye-Huckle constant function kappa
```

```
kappa_s = sqrt(I*8.486902807);
kappa_m = 0;

%2D staggered grid
xgrid = [a a+dx/2 : dx : b-dx/2 b];
ygrid = [c c+dy/2 : dy : d-dy/2 d];
[X, Y] = meshgrid(xgrid,ygrid);

%Initial location of each atom in water H20 (located on the grids)
%zi is Merz-Kollman ESP net atomic charges obtained from Q-Chem
x_H1 = 0.5000000;
y_H1 = 0.7400000;
zi_H1 = 0.426725;

x_O = 0.0200000;
y_O = 0.0200000;
zi_O = -0.846350;

x_H2 =  0.5000000;
y_H2 = -0.7400000;
zi_H2 = 0.419625;

x = [x_H1, x_O, x_H2];
y = [y_H1, y_O, y_H2];
z = [zi_H1, zi_O, zi_H2];

%RBF Interpolation (Gaussian RBF)
sigma = 0.0015;                  %parameter of the spread of the function
rbf = @(r) exp(-(r/(sigma)).^2); %r is a distance between 2 points

%RBF matrix (distance matrix)
R = zeros(length(x),length(x));
for i = 1:length(x)
    for j = 1:length(x)
    %for each pair of data points, calculate the Euclidean distance and apply rbf function
        R(i, j) = rbf(sqrt((x(i) - x(j))^2 + (y(i) - y(j))^2));
    end
end

%Solve linear system to find the weights for RBF interpolation
weights = R \ z';

%Interpolate at new points
Z = zeros(size(X));
for i = 1:numel(X)
    for j = 1:length(x)
        Z(i) = (Z(i) + weights(j) * rbf(sqrt((X(i) - x(j))^2 + (Y(i) - y(j))^2)));
    end
end

%RHS
rho = C.*Z;
rho = reshape(rho,[],1);

%Use PartitionOfUnity2DVectorFields for constant function epsilon
[epsilonh,epsilonv] = PartitionOfUnity2Dvector(4,1,epsilon_s,epsilon_m,m);
epsilonh = reshape(epsilonh,[],1);
epsilonv = reshape(epsilonv,[],1);
epsilon_hv = [epsilonh; epsilonv];
Epsilon = diag(epsilon_hv);

%Use PartitionOfUnity2DScalarFields for constant function kappa
kappa = (PartitionOfUnity2Dscalar(4,1,kappa_s,kappa_m,m).^2); %kappa^2
kappa = reshape(kappa,[],1);
Kappa = diag(kappa);

%Using the first term in the series expansion sinh(x) = x+x^3/3!+x^5/5!+...
%as an approximation of sinh(x). Therefore, mimetic differences would become
```

```
% –D(E (GU)) + K U = rho
% (–D(EG) + K) * U = rho
%Let A = (–D(EG) + K)
%             A * U = rho

%2D Mimetic gradient operator
G = grad2D(k,m,dx,n,dy);

%2D Mimetic divergence operator
D = div2D(k,m,dx,n,dy);

%Neumann BC
BC = robinBC2D(k, m, dx, n, dy, 0, 1);
A = –D*(Epsilon*G) + Kappa + BC;

% Solve for initial Ui
Ui = A \ rho;
Ui = reshape(Ui, m+2, n+2);
figure(1)
surf(X, Y, Ui);
xlabel('x');
ylabel('y');
zlabel('u_{i}(r)');

%Use Newton's Method to approximate a system of equations
%f(U) = –D(E(GU)) + K sinh(U) – rho
f = @(U) –D*(Epsilon*(G*U)) + Kappa*sinh(U) – rho;

Ui = reshape(Ui,[],1);
U = Ui;
maxIter = 100;
tolerance = 1e–15; %can be changed accordingly
iter = 0;

%Jacobian matrix is 1st partial derivatives of f with respect to U
J_linear = –D*(Epsilon*G);
J_nonlinear = diag(Kappa * cosh(U));
J = J_linear + J_nonlinear;
converge = false;
while iter < maxIter
    F = f(U);
    U_new = U – J\F;
    deltaU = U_new – U;
    if all(abs(deltaU) < tolerance) %check for convergence
        converge = true;
        break;
    end
    U = U_new;
    iter = iter + 1;
end

if converge
    disp(['Converged after ', num2str(iter), ' iterations.']);
else
    disp('Did not converge.');
end

U = reshape(U, m+2, n+2);
figure(2)
surf(X,Y,U);
xlabel('x');
ylabel('y');
zlabel('u(r)');
```

The input files of HCl molecule, SO_3 molecule, and HCN molecule are shown respectively below. Notice that all molecules are symmetrized and the atoms are shifted to be on the grid points

```
$molecule
0 1
Cl   -1.080000    1.4000000    0.0000000
H    0.3600000    1.3200000    0.0000000
$end

$rem
   BASIS  =  6-31g(D)
   ESP_CHARGES  =  TRUE
   GUI  =  2
   METHOD  =  B3LYP
   SCF_CONVERGENCE  =  8
$end
```

```
$molecule
0 1
S    0.0000000    0.0000000   -0.0000000
O    0.7678653   -1.3299818    0.0000000
O    0.7678653    1.3299818   -0.0000000
O   -1.5357307    0.0000000   -0.0000000
$end

$rem
BASIS  =  6-31g(D)
ESP_CHARGES  =  TRUE
GUI  =  2
METHOD  =  B3LYP
SCF_CONVERGENCE  =  8
$end
```

```
$molecule
0 1
C    0.0000000   -0.1005373    0.7324253
N    0.0000000    0.0206644   -0.7135753
H    0.0000000    0.9108826    1.1882759
$end

$rem
BASIS  =  6-31g(D)
ESP_CHARGES  =  TRUE
GUI  =  2
METHOD  =  B3LYP
SCF_CONVERGENCE  =  8
$end
```

The Merz-Kollman electrostatic potential net atomic charges of HCl,SO_3, and HCN molecule are listed in order below

Merz-Kollman ESP Net Atomic Charges

Atom	Charge (a.u.)
1 Cl	-0.247169
2 H	0.247169
Sum of atomic charges =	0.000000

Merz-Kollman ESP Net Atomic Charges

Atom	Charge (a.u.)
1 S	1.031620
2 O	-0.342988
3 O	-0.344316
4 O	-0.344316
Sum of atomic charges =	0.000000

Merz-Kollman ESP Net Atomic Charges

Atom	Charge (a.u.)
1 C	-0.109953
2 N	-0.146483
3 H	0.256436
Sum of atomic charges =	0.000000