Theoretical Foundations and Simulation of Diffusion Processes by use of Finite Differences

Materials Simulation Practical No.1

Anastasia Papadaki

Dr. Frank Wendler

WW8- Institute

Friedrich Alexander University Erlangen – Nuremberg

August, 2022

1. Introduction

Transport phenomena in nature can be distinguished according to whether transport occurs through directed, collective movement of particles or whether it is governed by their random movement. The former phenomena are called convection or advection, while the latter are denoted as diffusion.

If we take a look at the atomistic scale, then diffusion is the result of the non-directed motion of individual particles, driven by thermal fluctuations. If there exists a difference in concentration, statistically more particles move from higher to lower concentration than the other way. As a result, inhomogeneous materials (w.r.t. the atom types) can become homogeneous by diffusion. For diffusion to occur, the temperature should be high enough to overcome energy barriers to atomic motion.

In this practical we will take a different approach to diffusion, the phenomenological approach starting with Fick's laws of diffusion and their mathematical consequences: according to Fick's laws, the diffusion flux is proportional to the negative gradient of concentration. It goes from regions of higher concentration to regions of lower concentration.

This practical is divided into three main parts:

1) derivation of the governing partial differential equations
2) their numerical implementation for the special case of stationary diffusion,
3) interpretation/verification of the obtained results.

Task 1.1

Let Ω any arbitrary fixed region in the domain occupied by the material under consideration. The total mass (in moles) of the diffusing species in Ω at time t reads $m(t) = \int_{\Omega} \rho \, dV$, where $\rho = \rho(\vec{x}, t)$ denotes the concentration of the diffusing species at the position $\vec{x}$. When there is no source or sink process (i.e. absence of any chemical reaction), the mass conservation law implies that the rate of mass change in Ω equals to the net flow rate of the species across its boundary $\partial\Omega$, i.e.,

$$\frac{dm}{dt} = -\int_{\partial\Omega} \vec{j} \cdot \vec{n} \, dS \quad \Rightarrow \quad \frac{d}{dt}\left(\int_{\Omega} \rho \, dV \right) = -\int_{\partial\Omega} \vec{j} \cdot \vec{n} \, dS \,, \tag{1.1}$$

where $\vec{j} = \vec{j}(\vec{x}, t)$ is the diffusion flux (moles per unit area and time) of the species, and $\vec{n} = \vec{n}(\vec{x})$ is the outward unit vector normal to $\partial\Omega$.

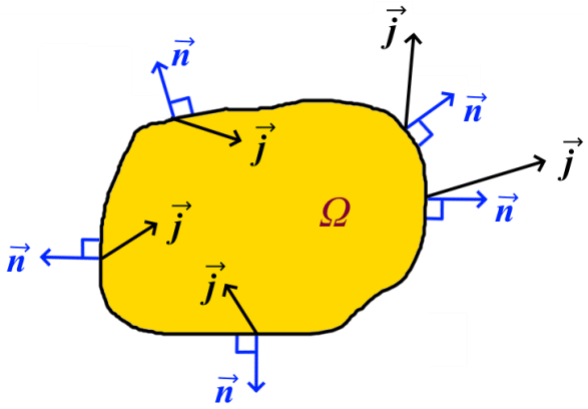

Figure 1: Rough drawing of an arbitrary control volume

Since Ω is fixed, the left-hand side of Eq. (1.1) equals to $\int_{\Omega} \frac{\partial \rho}{\partial t} dV$, while with the aid of the divergence theorem the surface integral on right-hand side of Eq. (1.1) obtains the form

$$\int_{\partial\Omega} \vec{\boldsymbol{j}} \cdot \vec{\boldsymbol{n}}\, dS = \int_{\Omega} \nabla \cdot \vec{\boldsymbol{j}}\, dV\,, \tag{1.2}$$

where $\nabla\cdot$ denotes the divergence. Therefore Eq. (1.1) can be recast into the form

$$\int_{\Omega} \frac{\partial \rho}{\partial t} dV = -\int_{\Omega} \nabla \cdot \vec{\boldsymbol{j}}\, dV \quad \Rightarrow \quad \int_{\Omega} \left(\frac{\partial \rho}{\partial t} + \nabla \cdot \vec{\boldsymbol{j}} \right) dV = 0\,. \tag{1.3}$$

Since this equation holds for any arbitrary region Ω, it follows that

$$\frac{\partial \rho(\vec{\boldsymbol{x}},t)}{\partial t} + \nabla \cdot \vec{\boldsymbol{j}}(\vec{\boldsymbol{x}},t) = 0\,, \tag{1.4}$$

which is the so-called *continuity equation* (i.e. local form of mass balance).

Fick's first law relates the diffusion flux $\vec{\boldsymbol{j}}(\vec{\boldsymbol{x}},t)$ to the gradient of the concentration $\nabla\rho(\vec{\boldsymbol{x}},t)$. It postulates that the flux goes from regions of high concentration to regions of low concentration, with a magnitude that is proportional to the concentration gradient, i.e.

$$\vec{\boldsymbol{j}}(\vec{\boldsymbol{x}},t) = -D \nabla \rho(\vec{\boldsymbol{x}},t)\,, \tag{1.5}$$

where in the simplest case of an isotropic and homogeneous medium, the diffusion coefficient D is a scalar material constant. In isotropic but non-homogeneous media, it is a scalar field, i.e. $D = D(\vec{\boldsymbol{x}})$, while in the most general case of an inhomogeneous anisotropic medium, D depends also in direction, i.e. it is a symmetric and positive definite tensor field $D = \boldsymbol{D}(\vec{\boldsymbol{x}})$.

Substituting Eq. (1.5) into (1.4), the most general form of the Fick's second law follows as

$$\frac{\partial \rho(\vec{\boldsymbol{x}},t)}{\partial t} = \nabla \cdot \left(\boldsymbol{D}(\vec{\boldsymbol{x}}) \nabla \rho(\vec{\boldsymbol{x}},t) \right). \tag{1.6}$$

It is noted that the above analysis is valid both in 3-D and 2-D. In 2-D, we set $\vec{\boldsymbol{x}} = (x_1, x_2)^T$ where x_1, x_2 are space coordinates in an arbitrary Cartesian system Ox_1x_2. Then,

$$\begin{aligned} \boldsymbol{D}(\vec{\boldsymbol{x}}) \nabla \rho(\vec{\boldsymbol{x}},t) &= \begin{bmatrix} D_{11}(x_1,x_2) & D_{12}(x_1,x_2) \\ D_{12}(x_1,x_2) & D_{22}(x_1,x_2) \end{bmatrix} \begin{bmatrix} \dfrac{\partial \rho(x_1,x_2,t)}{\partial x_1} \\ \dfrac{\partial \rho(x_1,x_2,t)}{\partial x_2} \end{bmatrix} = \\ &= \begin{bmatrix} D_{11}(x_1,x_2) \dfrac{\partial \rho(x_1,x_2,t)}{\partial x_1} + D_{12}(x_1,x_2) \dfrac{\partial \rho(x_1,x_2,t)}{\partial x_2} \\ D_{12}(x_1,x_2) \dfrac{\partial \rho(x_1,x_2,t)}{\partial x_1} + D_{22}(x_1,x_2) \dfrac{\partial \rho(x_1,x_2,t)}{\partial x_2} \end{bmatrix} \end{aligned} \tag{1.7}$$

and thus (for simplicity the arguments (x_1, x_2, t) and (x_1, x_2) are not written),

$$\nabla\cdot\left(\boldsymbol{D}(\vec{\boldsymbol{x}})\nabla\rho(\vec{\boldsymbol{x}},t)\right)=\frac{\partial}{\partial x_1}\left(D_{11}\frac{\partial\rho}{\partial x_1}+D_{12}\frac{\partial\rho}{\partial x_2}\right)+\frac{\partial}{\partial x_2}\left(D_{12}\frac{\partial\rho}{\partial x_1}+D_{22}\frac{\partial\rho}{\partial x_2}\right)=$$

$$=D_{11}\frac{\partial^2\rho}{\partial x_1^2}+2D_{12}\frac{\partial^2\rho}{\partial x_1\partial x_2}+D_{22}\frac{\partial^2\rho}{\partial x_2^2}+\left(\frac{\partial D_{11}}{\partial x_1}+\frac{\partial D_{12}}{\partial x_2}\right)\frac{\partial\rho}{\partial x_1}+\left(\frac{\partial D_{12}}{\partial x_1}+\frac{\partial D_{22}}{\partial x_2}\right)\frac{\partial\rho}{\partial x_2}= \tag{1.8}$$

$$=\sum_{i,j=1}^{2}D_{ij}\frac{\partial^2\rho}{\partial x_i\partial x_j}+\sum_{i,j=1}^{2}\frac{\partial D_{ij}}{\partial x_i}\frac{\partial\rho}{\partial x_j}$$

Accordingly, Fick's second law obtains the form

$$\frac{\partial\rho}{\partial t}=\sum_{i,j=1}^{2}D_{ij}\frac{\partial^2\rho}{\partial x_i\partial x_j}+\sum_{i,j=1}^{2}\frac{\partial D_{ij}}{\partial x_i}\frac{\partial\rho}{\partial x_j} \tag{1.9}$$

In the case of a homogeneous but anisotropic medium, the tensor $\boldsymbol{D}$ is constant ($\partial D_{ij}/\partial x_i=0$) and thus, Eq. (1.9) is simplified as

$$\frac{\partial\rho}{\partial t}=\sum_{i,j=1}^{2}D_{ij}\frac{\partial^2\rho}{\partial x_i\partial x_j}\quad\Rightarrow\quad\frac{\partial\rho}{\partial t}=D_{11}\frac{\partial^2\rho}{\partial x_1^2}+2D_{12}\frac{\partial^2\rho}{\partial x_1\partial x_2}+D_{22}\frac{\partial^2\rho}{\partial x_2^2} \tag{1.10}$$

Changing notation from x_1, x_2 to x, y, this can also be written as

$$\frac{\partial\rho}{\partial t}=D_{xx}\frac{\partial^2\rho}{\partial x^2}+2D_{xy}\frac{\partial^2\rho}{\partial x\partial y}+D_{yy}\frac{\partial^2\rho}{\partial y^2}. \tag{1.11}$$

The coefficient matrix of the 2nd order derivatives with respect x, y and t reads $A=\begin{bmatrix}D_{xx} & D_{xy} & 0\\ D_{xy} & D_{yy} & 0\\ 0 & 0 & 0\end{bmatrix}$. This has a zero eigenvalues along with two positive eigenvalues (the eigenvalues of the symmetric and positive definite tensor $\boldsymbol{D}$). Therefore, Eq. (1.11) is **parabolic**. The same holds true also for the more general Eq. (1.9) since it has the same coefficient matrix.

Task 1.2

Since $\boldsymbol{D}$ is symmetric, it is also diagonizable. In fact, there is an orhonormal basis consisting of unit eigenvectors of $\boldsymbol{D}$, and $\boldsymbol{D}$ is diagonal with respect this basis. In other words, there is a Cartesian system Oxy (system of principal axes of $\boldsymbol{D}$) such that $\boldsymbol{D}$ is diagonal ($D_{xy}=0$) with respect this system. In this Cartesian system, Eq. (1.11) obtains the form

$$\frac{\partial\rho}{\partial t}=D_{xx}\frac{\partial^2\rho}{\partial x^2}+D_{yy}\frac{\partial^2\rho}{\partial y^2}. \tag{1.12}$$

For stationary diffusion $\frac{\partial\rho}{\partial t}=0$, and hence (1.12) gives $D_{xx}\partial_{xx}^2\rho+D_{yy}\partial_{yy}^2\rho=0\Rightarrow$ $\Rightarrow\partial_{xx}^2\rho+\frac{D_{yy}}{D_{xx}}\partial_{yy}^2\rho=0$, where $\rho=\rho(x,y)$. Therefore,

$$\partial^2_{xx}\rho(x,y) + P\,\partial^2_{yy}\rho(x,y) = 0\,, \qquad P := \frac{D_{yy}}{D_{xx}}\,. \tag{1.13}$$

In this expression x, y are Cartesian coordinates with respect the principal axes of the diffusion tensor $\boldsymbol{D}$.

Task 1.3

Using the linear transformation $\tilde{x} = x$, $\tilde{y} = \frac{y}{\sqrt{P}}$ and defining $\tilde{\rho}(\tilde{x},\tilde{y}) = \tilde{\rho}(x, y/\sqrt{P}) := \rho(x,y)$, we have $\partial^2_{xx}\rho(x,y) = \partial^2_{\tilde{x}\tilde{x}}\tilde{\rho}(\tilde{x},\tilde{y})$, $\partial_y\rho(x,y) = \partial_{\tilde{y}}\tilde{\rho}(\tilde{x},\tilde{y})\frac{\partial\tilde{y}}{\partial y} = \partial_{\tilde{y}}\tilde{\rho}(\tilde{x},\tilde{y})\frac{1}{\sqrt{P}}$, and thus, $\partial_{yy}\rho(x,y) = \partial_{\tilde{y}\tilde{y}}\tilde{\rho}(\tilde{x},\tilde{y})\left(\frac{1}{\sqrt{P}}\right)^2 = \frac{1}{P}\partial_{\tilde{y}\tilde{y}}\tilde{\rho}(\tilde{x},\tilde{y})$. Accordingly, Eq. (1.13) can be written as

$$\partial^2_{\tilde{x}\tilde{x}}\tilde{\rho}(\tilde{x},\tilde{y}) + \partial^2_{\tilde{y}\tilde{y}}\tilde{\rho}(\tilde{x},\tilde{y}) = 0\,, \tag{1.14}$$

which is a Laplace equation (which is an elliptic equation).

Note that stationary heat transfer has the same PDE structure and can be treated in full analogy.

Task 2.1

Numerical solutions of Eq. $(1.13)_1$ in rectangular domains of the x, y-plane can be obtained using finite difference method. In particular, the rectangular domain of interest is discretized by a equidistant mesh of nodes (i, j) at which the function $\rho(x,y)$ will be evaluated. The node distances are h_x and h_y in the x and y directions respectively, as shown in Figure 2.

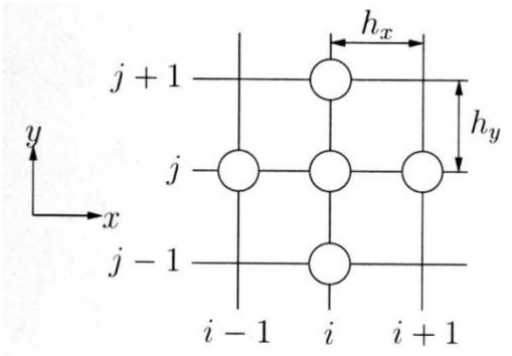

Figure 2: Small section of the discretize x, y-domain: circles denote the nodes that are used for the finite difference scheme at node (i, j)

Using a second order accurate central difference scheme, the 1st and 2nd partial derivatives of $\rho(x,y)$ at the point (x,y) are approximated as

$$\frac{\partial\rho(x,y)}{\partial x} \approx \frac{\rho(x+h_x,y)-\rho(x-h_x,y)}{2h_x}, \qquad \frac{\partial\rho(x,y)}{\partial y} \approx \frac{\rho(x,y+h_y)-\rho(x,y-h_y)}{2h_y} \tag{2.1}$$

$$\frac{\partial^2\rho(x,y)}{\partial x^2} \approx \frac{\rho(x+h_x,y)-2\rho(x,y)+\rho(x-h_x,y)}{h_x^2} \tag{2.2}$$

$$\frac{\partial^2\rho(x,y)}{\partial y^2} \approx \frac{\rho(x,y+h_y)-2\rho(x,y)+\rho(x,y-h_y)}{h_y^2} \tag{2.3}$$

Accordingly, if $\rho_{i,j}$ denotes the numerical (i.e. approximate) value of $\rho(x,y)$ at the node (i,j), then the discretized counterpart of Eq. $(1.13)_1$ reads

$$\frac{\rho_{i+1,j}-2\rho_{i,j}+\rho_{i-1,j}}{h_x^2} + P\frac{\rho_{i,j+1}-2\rho_{i,j}+\rho_{i,j-1}}{h_y^2} = 0. \tag{2.4}$$

Task 2.2

a) Let a 2-dimensional rectangular region whose boundaries of the rectangle are aligned with the principal axes of the diffusion tensor. The region is first discretized by 5×5 nodes which are numbered row-wise from top left (nodes 1) to bottom right (node 25), as shown in Figure 3.

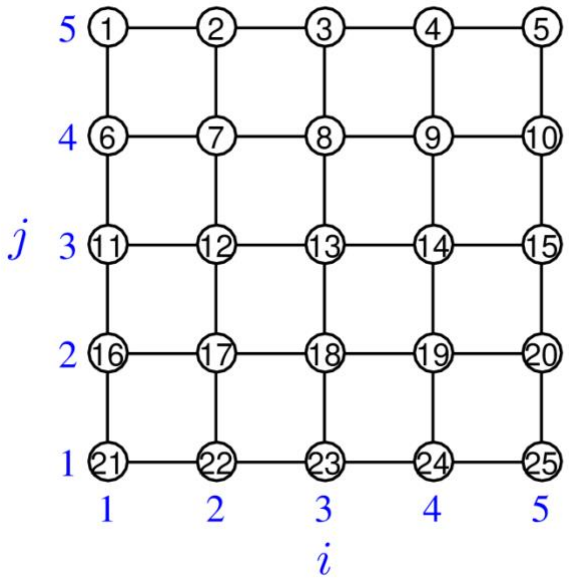

Figure 3: Nodes numbering in a 5x5 nodes discretization. The respective (i,j) numbers are also depicted in blue color

The mapping from the (i,j) notation to the consecutive node-numbering notation of a general system with $N\times M$ nodes reads

$$\begin{bmatrix} (1,M) & (2,M) & \cdots & (N,M) \\ (1,M-1) & (2,M-1) & \cdots & (N,M-1) \\ \vdots & \vdots & \vdots & \vdots \\ (1,2) & (2,2) & \cdots & (N,2) \\ (1,1) & (2,1) & \cdots & (N,1) \end{bmatrix} \mapsto \begin{bmatrix} 1 & 2 & \cdots & N \\ N+1 & N+2 & \cdots & 2N \\ \vdots & \vdots & \vdots & \vdots \\ (M-2)N+1 & (M-2)N+2 & \cdots & (M-1)N \\ (M-1)N+1 & (M-1)N+2 & \cdots & MN \end{bmatrix}$$

i.e., the numbering of nodes changes from the pair (i, j), with $i = 1, 2, \ldots N$ and $j = 1, 2, \ldots M$ to a single number k, with $i = 1, 2, \ldots MN$ where $m = MN$. From this matrix-like mapping, it can be easily found that the mapping $(i, j) \mapsto k$ is described by the relation $k = (M - j)N + i$.

b) From the above mapping $(i, j) \mapsto k$, we have that

$$\left.\begin{aligned} (i+1, j) &\mapsto k+1 \\ (i-1, j) &\mapsto k-1 \\ (i, j+1) &\mapsto k-N \\ (i, j-1) &\mapsto k+N \end{aligned}\right\} \tag{2.5}$$

Using the single index notation, i.e. $\rho_k \coloneqq \rho_{i,j}$, Eq. (2.4) obtains the form

$$\frac{\rho_{k+1} - 2\rho_k + \rho_{k-1}}{h_x^2} + P\,\frac{\rho_{k-N} - 2\rho_k + \rho_{k+N}}{h_y^2} = 0, \tag{2.6}$$

which, after rearranging terms, can also be written as

$$\frac{P}{h_y^2}\rho_{k-N} + \frac{1}{h_x^2}\rho_{k-1} - 2\left(\frac{1}{h_x^2} + \frac{P}{h_y^2}\right)\rho_k + \frac{1}{h_x^2}\rho_{k+1} + \frac{P}{h_y^2}\rho_{k+N} = 0. \tag{2.7}$$

For the 5×5 discretization and for the internal nodes (i.e., for $k = 7,8,9,12,13,14,17,18,19$), this gives 9×25 linear system (9 equations for the 9 different values of k, and 25 unknowns ρ_l, $l = 1, \ldots, 25$). In particular, we have

$$
\left.
\begin{aligned}
&\frac{P}{h_y^2}\rho_2+\frac{1}{h_x^2}\rho_6-2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right)\rho_7+\frac{1}{h_x^2}\rho_8+\frac{P}{h_y^2}\rho_{12}=0\\
&\frac{P}{h_y^2}\rho_3+\frac{1}{h_x^2}\rho_7-2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right)\rho_8+\frac{1}{h_x^2}\rho_9+\frac{P}{h_y^2}\rho_{13}=0\\
&\frac{P}{h_y^2}\rho_4+\frac{1}{h_x^2}\rho_8-2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right)\rho_9+\frac{1}{h_x^2}\rho_{10}+\frac{P}{h_y^2}\rho_{14}=0\\
&\frac{P}{h_y^2}\rho_7+\frac{1}{h_x^2}\rho_{11}-2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right)\rho_{12}+\frac{1}{h_x^2}\rho_{13}+\frac{P}{h_y^2}\rho_{17}=0\\
&\frac{P}{h_y^2}\rho_8+\frac{1}{h_x^2}\rho_{12}-2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right)\rho_{13}+\frac{1}{h_x^2}\rho_{14}+\frac{P}{h_y^2}\rho_{18}=0\\
&\frac{P}{h_y^2}\rho_9+\frac{1}{h_x^2}\rho_{13}-2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right)\rho_{14}+\frac{1}{h_x^2}\rho_{15}+\frac{P}{h_y^2}\rho_{19}=0\\
&\frac{P}{h_y^2}\rho_{12}+\frac{1}{h_x^2}\rho_{16}-2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right)\rho_{17}+\frac{1}{h_x^2}\rho_{18}+\frac{P}{h_y^2}\rho_{22}=0\\
&\frac{P}{h_y^2}\rho_{13}+\frac{1}{h_x^2}\rho_{17}-2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right)\rho_{18}+\frac{1}{h_x^2}\rho_{19}+\frac{P}{h_y^2}\rho_{23}=0\\
&\frac{P}{h_y^2}\rho_{14}+\frac{1}{h_x^2}\rho_{18}-2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right)\rho_{19}+\frac{1}{h_x^2}\rho_{20}+\frac{P}{h_y^2}\rho_{24}=0
\end{aligned}
\right\} \qquad (2.8)
$$

From this, it follows that the inner part of the coefficient matrix, i.e. the part relating internal nodes to internal nodes (nodes $7,8,9,12,13,14,17,18,19$), can be partitioned into nine 3×3 blocks: $\begin{bmatrix} B & J & O \\ J & B & J \\ O & J & B \end{bmatrix}$, where O denotes zero submatrices, and

$$
B=\begin{bmatrix}
-2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right) & \frac{1}{h_x^2} & 0 \\
\frac{1}{h_x^2} & -2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right) & \frac{1}{h_x^2} \\
0 & \frac{1}{h_x^2} & -2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right)
\end{bmatrix}, \qquad J=\left(\frac{P}{h_y^2}\right)I_3, \qquad (2.9)
$$

with I_3 denoting the 3×3 identity matrix.

For the general $N\times M$ discretization, the number of internal nodes is $(N-2)(M-2)$ and the inner part of the coefficient matrix can be partitioned into blocks as

$$\begin{bmatrix} B & J & & \\ J & B & \ddots & \\ & \ddots & \ddots & J \\ & & J & B \end{bmatrix}$$

where the shape of submatrices B, J is $(N-2)\times(N-2)$ and the submatrix B appears $M-2$ times. Moreover,

$$B = \begin{bmatrix} -2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right) & \frac{1}{h_x^2} & & \\ \frac{1}{h_x^2} & -2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right) & \ddots & \\ & \ddots & \ddots & \frac{1}{h_x^2} \\ & & \frac{1}{h_x^2} & -2\left(\frac{1}{h_x^2}+\frac{P}{h_y^2}\right) \end{bmatrix}, \qquad J = \left(\frac{P}{h_y^2}\right) I_{N-2} \tag{2.10}$$

i.e. B is a $(N-2)\times(N-2)$ tridiagonal submatrix.

For boundary nodes, Eq. (2.4) and therefore, (2.6) and (2.7) cannot be valid in this form, since some of the indices appeared are out of range. For example, for $k = 1,\ldots,N$, the index $k-N$ in Eq. (2.6) becomes non-positive. In fact, for boundary nodes, boundary conditions must be taken into account, as shown in the following tasks.

It is also noted that in the following, we will consider only the **isotropic** case, i.e. $D_{yy} = D_{xx} =: D$ and therefore, $P = 1$.

In the next, we assume that the top and lower boundary of the system have the concentration (or temperature in the case of stationary heat transfer) prescribed through Dirichlet boundary conditions, while at the left and right boundary, the diffusion flux is prescribed, i.e. von Neumann boundary conditions are used (see Figure 4).

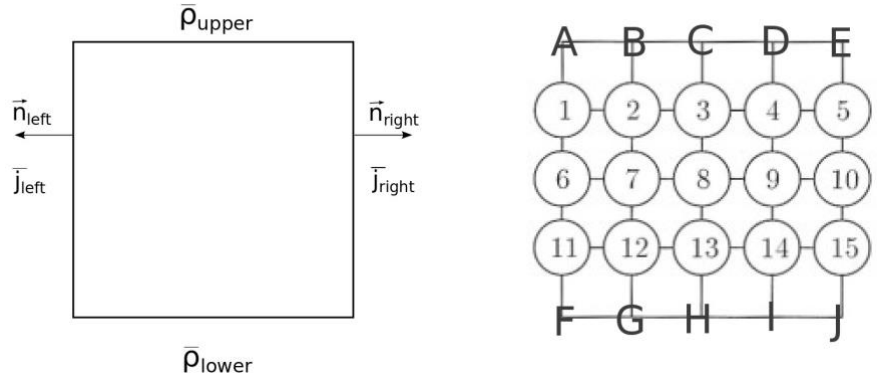

Figure 4: System and discretization

Task 3.1: Dirichlet boundary conditions

For example, the numerical evaluation of the governing PDE at node 3 (Figure 4) requires the value of ρ at nodes 2,3,4,8 and C. In particular, from Eq. (2.6) and for the isotropic case ($P=1$), we obtain

$$\frac{\rho_4-2\rho_3+\rho_2}{h_x^2}+\frac{\rho_C-2\rho_3+\rho_8}{h_y^2}=0\Rightarrow\frac{1}{h_x^2}\rho_2-2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_3+\frac{1}{h_x^2}\rho_4+\frac{1}{h_y^2}\rho_8=-\frac{1}{h_y^2}\rho_C\,, \qquad (3.1)$$

Task 3.2: Neumann boundary conditions

For the left and right boundary we assume von Neumann boundary conditions, i.e. $\vec{\boldsymbol{j}}\cdot\vec{\boldsymbol{n}}$ obtains the prescribed values $\bar{j}_{\text{left}}$ and $\bar{j}_{\text{right}}$, respectively. Since $\vec{\boldsymbol{n}}$ is the outward normal to the boundary, $\bar{j}_{\text{left}}$ and $\bar{j}_{\text{right}}$ are negative if the flux is directed into the domain and they are positive if the flux is directed outwards.

From Fick's first law we have $\vec{\boldsymbol{j}}=-D\nabla\rho$, and hence the boundary conditions $\vec{\boldsymbol{j}}\cdot\vec{\boldsymbol{n}}_{\text{left/right}}=\bar{j}_{\text{left/right}}$ obtain the form $-D\nabla\rho\cdot\vec{\boldsymbol{n}}_{\text{left/right}}=\bar{j}_{\text{left/right}}\Rightarrow\nabla\rho\cdot\vec{\boldsymbol{n}}_{\text{left/right}}=-\dfrac{\bar{j}_{\text{left/right}}}{D}$, i.e.,

$$\left.\frac{\partial\rho(x,y)}{\partial x}\right|_{\text{left}}=\frac{\bar{j}_{\text{left}}}{D}\,,\qquad\left.\frac{\partial\rho(x,y)}{\partial x}\right|_{\text{right}}=-\frac{\bar{j}_{\text{right}}}{D} \qquad (3.2)$$

The corresponding discretized form of Eq. (3.2) is derived by using the approximations given in Eq. $(2.1)_1$. In particular, the first derivative w.r.t x at an internal node k is approximated as

$$\frac{\partial\rho(x,y)}{\partial x}\cong\frac{\rho_{k+1}-\rho_{k-1}}{2h_x} \qquad (3.3)$$

where $k-1$ and $k+1$ are the left and right neighbor nodes of k. When k is on the left boundary, $k-1$ node **is not** the left neighbor of k and a left "ghost node" (denoted, e.g. as "*kl*") must be introduced. In a similar manner, if k is on the right boundary, $k+1$ node **is not** the right neighbor of k and a right "ghost node" (numbered, e.g. as "*kr*") must be introduced. Accordingly, the boundary conditions (3.2) have the discretized form

$$\frac{\rho_{k+1}-\rho_{kl}}{2h_x}=\frac{\bar{j}_{\text{left}}}{D}\,,\qquad\frac{\rho_{kr}-\rho_{k-1}}{2h_x}=-\frac{\bar{j}_{\text{right}}}{D} \qquad (3.4)$$

These can be solved w.r.t. ρ_{kl},ρ_{kr}, i.e.

$$\rho_{kl}=\rho_{k+1}-\frac{\bar{j}_{\text{left}}}{D}2h_x\,,\qquad\rho_{kr}=\rho_{k-1}-\frac{\bar{j}_{\text{right}}}{D}2h_x \qquad (3.5)$$

Then, for a node k on the left boundary, Eq. (2.6) gives ($k-1$ is replaced by kl)

$$\frac{\rho_{k+1}-2\rho_k+\rho_{kl}}{h_x^2}+\frac{\rho_{k-N}-2\rho_k+\rho_{k+N}}{h_y^2}=0 \quad \overset{(3.5)_1}{\Rightarrow}$$

$$\Rightarrow \quad \frac{\rho_{k+1}-2\rho_k+\rho_{k+1}-\dfrac{\bar{j}_{\text{left}}}{D}2h_x}{h_x^2}+\frac{\rho_{k-N}-2\rho_k+\rho_{k+N}}{h_y^2}=0 \Rightarrow$$

$$\Rightarrow \quad \frac{1}{h_y^2}\rho_{k-N}-2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_k+\frac{2}{h_x^2}\rho_{k+1}+\frac{1}{h_y^2}\rho_{k+N}=2\frac{\bar{j}_{\text{left}}}{h_xD} \qquad (k \text{ on left boundary}) \qquad (3.6)$$

It is also noted that for $k=1$, we have $\rho_{k-N}=\rho_A$ and thus, the first term should be moved on the right hand side (as $-\rho_A/h_y^2$). In a similar manner, if k is on the bottom left corner just above the lower boundary ($k=11$ in Figure 4), then $\rho_{k+N}=\rho_F$ and now, the last term should be move on the right hand side (as $-\rho_F/h_y^2$).

For a node k on the right boundary, Eq. (2.6) gives ($k+1$ is replaced by kr)

$$\frac{\rho_{kr}-2\rho_k+\rho_{k-1}}{h_x^2}+\frac{\rho_{k-N}-2\rho_k+\rho_{k+N}}{h_y^2}=0 \quad \overset{(3.5)_2}{\Rightarrow}$$

$$\Rightarrow \quad \frac{\rho_{k-1}-\dfrac{\bar{j}_{\text{right}}}{D}2h_x-2\rho_k+\rho_{k-1}}{h_x^2}+\frac{\rho_{k-N}-2\rho_k+\rho_{k+N}}{h_y^2}=0 \Rightarrow$$

$$\Rightarrow \quad \frac{1}{h_y^2}\rho_{k-N}+\frac{2}{h_x^2}\rho_{k-1}-2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_k+\frac{1}{h_y^2}\rho_{k+N}=2\frac{\bar{j}_{\text{right}}}{h_xD} \qquad (k \text{ on right boundary}) \qquad (3.7)$$

For example, for node 10 in Figure 4, Eq. (3.7) gives

$$\frac{1}{h_y^2}\rho_5+\frac{2}{h_x^2}\rho_9-2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_{10}+\frac{1}{h_y^2}\rho_{15}=2\frac{\bar{j}_{\text{right}}}{h_xD} \qquad (3.8)$$

It is also noted that for $k=N$ ($k=5$ in Figure 4), we have $\rho_{k-N}=\rho_E$ and thus, the first term on l.h.s. of Eq. (3.7) should be moved on the right hand side (as $-\rho_E/h_y^2$). In a similar manner, if k is on the bottom right corner just above the lower boundary ($k=15$ in Figure 4), then $\rho_{k+N}=\rho_J$ and now, the last term on l.h.s. of Eq. (3.7) should be move on the right hand side (as $-\rho_J/h_y^2$).

Task 4.1: Assemble the Linear System of Equations

Applying Eq. (3.6) for the nodes on the left boundary, Eq. (3.7) for the nodes on the right boundary and Eq. (2.7) for the rest of the nodes and taking into account that $k-N, k+N, k-1, k+1$ denote the nearest neighbor nodes of node k above, below, left and right, respectively, we have

$$-2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_1+\frac{2}{h_x^2}\rho_2+\frac{1}{h_y^2}\rho_6=-\frac{1}{h_y^2}\rho_A+2\frac{\bar{j}_{\text{left}}}{h_xD}$$

$$\frac{1}{h_x^2}\rho_1-2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_2+\frac{1}{h_x^2}\rho_3+\frac{1}{h_y^2}\rho_7=-\frac{1}{h_y^2}\rho_B$$

$$\frac{1}{h_x^2}\rho_2 - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_3 + \frac{1}{h_x^2}\rho_4 + \frac{1}{h_y^2}\rho_8 = -\frac{1}{h_y^2}\rho_C$$

$$\frac{1}{h_x^2}\rho_3 - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_4 + \frac{1}{h_x^2}\rho_5 + \frac{1}{h_y^2}\rho_9 = -\frac{1}{h_y^2}\rho_D$$

$$\frac{2}{h_x^2}\rho_4 - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_5 + \frac{1}{h_y^2}\rho_{10} = -\frac{1}{h_y^2}\rho_E + 2\frac{\bar{j}_{\text{right}}}{h_x D}$$

$$\frac{1}{h_y^2}\rho_1 - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_6 + \frac{2}{h_x^2}\rho_7 + \frac{1}{h_y^2}\rho_{11} = 2\frac{\bar{j}_{\text{left}}}{h_x D}$$

$$\frac{1}{h_y^2}\rho_2 + \frac{1}{h_x^2}\rho_6 - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_7 + \frac{1}{h_x^2}\rho_8 + \frac{1}{h_y^2}\rho_{12} = 0$$

$$\frac{1}{h_y^2}\rho_3 + \frac{1}{h_x^2}\rho_7 - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_8 + \frac{1}{h_x^2}\rho_9 + \frac{1}{h_y^2}\rho_{13} = 0$$

$$\frac{1}{h_y^2}\rho_4 + \frac{1}{h_x^2}\rho_8 - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_9 + \frac{1}{h_x^2}\rho_{10} + \frac{1}{h_y^2}\rho_{14} = 0$$

$$\frac{1}{h_y^2}\rho_5 + \frac{2}{h_x^2}\rho_9 - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_{10} + \frac{1}{h_y^2}\rho_{15} = 2\frac{\bar{j}_{\text{right}}}{h_x D}$$

$$\frac{1}{h_y^2}\rho_6 - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_{11} + \frac{2}{h_x^2}\rho_{12} = \frac{1}{h_y^2}\rho_F + 2\frac{\bar{j}_{\text{left}}}{h_x D}$$

$$\frac{1}{h_y^2}\rho_7 + \frac{1}{h_x^2}\rho_{11} - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_{12} + \frac{1}{h_x^2}\rho_{13} = -\frac{1}{h_y^2}\rho_G$$

$$\frac{1}{h_y^2}\rho_8 + \frac{1}{h_x^2}\rho_{12} - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_{13} + \frac{1}{h_x^2}\rho_{14} = -\frac{1}{h_y^2}\rho_H$$

$$\frac{1}{h_y^2}\rho_9 + \frac{1}{h_x^2}\rho_{13} - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_{14} + \frac{1}{h_x^2}\rho_{15} = -\frac{1}{h_y^2}\rho_I$$

$$\frac{1}{h_y^2}\rho_{10} + \frac{2}{h_x^2}\rho_{14} - 2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)\rho_{15} = -\frac{1}{h_y^2}\rho_J + 2\frac{\bar{j}_{\text{right}}}{h_x D}$$

This linear system can be recast in the matrix form $Ax = b$, where the coefficient matrix can be partitioned into nine blocks as $A = \begin{bmatrix} B & J & O \\ J & B & J \\ O & J & B \end{bmatrix}$, where each submatrix has the shape 5×5. In particular, $J = \frac{1}{h_y^2} I_5$, where I_5 is the 5×5 identity matrix, and

$$B=\begin{bmatrix} -2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right) & \frac{2}{h_x^2} & 0 & 0 & 0 \\ \frac{1}{h_x^2} & -2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right) & \frac{1}{h_x^2} & 0 & 0 \\ 0 & \frac{1}{h_x^2} & -2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right) & \frac{1}{h_x^2} & 0 \\ 0 & 0 & \frac{1}{h_x^2} & -2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right) & \frac{1}{h_x^2} \\ 0 & 0 & 0 & \frac{2}{h_x^2} & -2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right) \end{bmatrix}. \qquad (4.1)$$

Moreover, $x=\begin{bmatrix}\rho_1\\ \rho_2\\ \vdots\\ \rho_{15}\end{bmatrix}$ is the vector of unknowns, while the column b is

$$b=\frac{2}{h_x D}\begin{bmatrix}\overline{j}_{\text{left}} & 0 & 0 & 0 & \overline{j}_{\text{right}} & \overline{j}_{\text{left}} & 0 & 0 & 0 & \overline{j}_{\text{right}} & \overline{j}_{\text{left}} & 0 & 0 & 0 & \overline{j}_{\text{right}}\end{bmatrix}^T + \\ -\frac{1}{h_y^2}\begin{bmatrix}\rho_A & \rho_B & \rho_C & \rho_D & \rho_E & 0 & 0 & 0 & 0 & 0 & \rho_F & \rho_G & \rho_H & \rho_I & \rho_J\end{bmatrix}^T \qquad (4.2)$$

Columns 1,6,11 of the matrix A correspond to the left boundary, columns 5,10,15 correspond to right boundary, while the rest of columns correspond to internal nodes. Since Dirichlet boundary conditions have been used for the top and lower boundaries, the values of ρ at them are known and they appear in the second vector of b (see Eq.(4.2)).

Task 4.2: Properties of Coefficient Matrix

As shown above, the coefficient matrix A is a ***block tridiagonal matrix*** of the form $A=\begin{bmatrix}B & J & O\\ J & B & J\\ O & J & B\end{bmatrix}$, i.e. it contains the tridiagonal square submatrix B on the main diagonal and the diagonal submatrix J on the subdiagonal and the supradiagonal. Accordingly, A is a ***sparse matrix*** whose non-zero entries are confined to ***five diagonals***. Accordingly, it is also ***banded***.

Each row in the block corresponds to the equations for the nodes in the same row of the discretization, i.e. $\begin{bmatrix}B & J & O\end{bmatrix}$ corresponds to the equations for the nodes $1,\ldots,5$, $\begin{bmatrix}J & B & J\end{bmatrix}$ corresponds to the equations for the nodes $6,\ldots,10$, and $\begin{bmatrix}O & J & B\end{bmatrix}$ corresponds to the equations for the nodes $11,\ldots,15$.

Furthermore, as shown in Eq. (4.1), submatrix B is a tridiagonal matrix with the same element $-2\left(\frac{1}{h_x^2}+\frac{1}{h_y^2}\right)$ on the main diagonal, while its subdiagonal and supradiagonal consist of only the

element $1/h_x^2$ except for the first element of supradiagonal and the last element of subdiagonal which is $2/h_x^2$ because of the Neumann boundary conditions. Moreover, the submatrix J is diagonal. In fact, it is proportional to the identity matrix, as mentioned above.

The coefficient matrix A is ***strictly diagonally dominant*** (and hence ***invertible***) since at each row the magnitude $2\left(1/h_x^2+1/h_y^2\right)$ of the diagonal entry is larger than the sum $2/h_x^2+1/h_y^2$ of the magnitudes of all the other (non-diagonal) entries.

The shape of the coefficient matrix A is influenced by the shape $N\times M$ of the discretization (N nodes in the direction x and M nodes in the direction y), as well as by the type of the boundary conditions assumed, because both determine the number of unknowns. For the boundary conditions assumed herein, the number of nodes where Dirichlet boundary conditions are used (top and lower boundary) is $2N$. Accordingly, the number of unknowns is $MN-2N=(M-2)N$, and therefore, A will be $(M-2)N\times(M-2)N$.

Task 5.1

The present finite difference scheme is implemented in Matlab. In particular, a function named "*FinDiff_Laplace_Eq*" is developed which is shown in Table A.1 in Appendix. The input arguments of this function are the width L_x and the height L_y, the number of nodes N_x and N_y (denoted by N and M, respectively in the previous sections) and the prescribed values of boundary conditions $\bar{\rho}_{\text{upper}}$, $\bar{\rho}_{\text{lower}}$ $\bar{j}_{\text{left}}/D$, $\bar{j}_{\text{right}}/D$. The flux boundary conditions have been normalized by the diffusion coefficient D because they appear in this form in vector b (cf. Eq. (4.2)).

The node distances are calculated as $h_x=L_x/(N_x-1)$ and $h_y=L_y/(N_y-1)$. The coefficient matrix A is first constructed by building a $(N_xN_y)\times(N_xN_y)$ matrix with the value $-2\left(1/h_x^2+1/h_y^2\right)$ on its main diagonal, the value $1/h_x^2$ in its first subdiagonal and supradiagonal, the value $1/h_y^2$ on the two diagonals having N_x positional distance from the main diagonal, and zeros elsewhere. Then, Neumann BCs are taken into account by setting the values $A_{k,k+1}=2/h_x^2$, $A_{k,k-1}=0$ for boundary nodes k where $\bar{j}_{\text{left}}/D$ is prescribed, and $A_{k,k+1}=0$, $A_{k,k-1}=2/h_x^2$ for boundary nodes k where $\bar{j}_{\text{right}}/D$ is prescribed. Finally, rows and columns corresponding to Dirichlet BCs are removed and the final shape of A becomes $(N_y-2)N_x\times(N_y-2)N_x$. A similar procedure is employed for the construction of the right hand side vector b.

It is noted that in all simulations presented below, the values $L_x=L_y=10$ have been used.

Task 5.2: Test Problems

1) For $\bar{j}_{\text{left}}=\bar{j}_{\text{right}}=0$ and $\bar{\rho}_{\text{upper}}=\bar{\rho}_{\text{lower}}=100\,\text{mol}\cdot\text{m}^{-3}$ the concentration is constant, i.e. $\rho(x,y)=100$ everywhere, because this is the solution of the Laplace equation $\partial_{xx}^2\rho(x,y)+\partial_{yy}^2\rho(x,y)=0$ that satisfies these BCs. The result is shown in Figure 5.

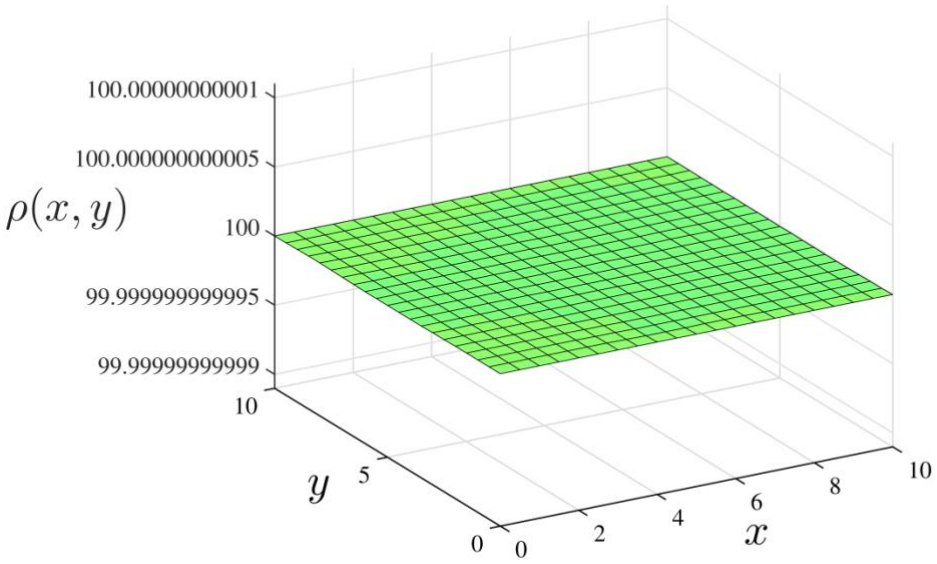

Figure 5: Numerical solution for $\bar{j}_{\text{left}} = \bar{j}_{\text{right}} = 0$, $\bar{\rho}_{\text{upper}} = \bar{\rho}_{\text{lower}} = 100\ \text{mol}\cdot\text{m}^{-3}$ obtained using a 21×21 grid.

2) For $\bar{j}_{\text{left}} = \bar{j}_{\text{right}} = 0$, $\bar{\rho}_{\text{upper}} = 100\ \text{mol}\cdot\text{m}^{-3}$ and $\bar{\rho}_{\text{lower}} = 50\ \text{mol}\cdot\text{m}^{-3}$ the concentration $\rho(x,y)$ is a linear function of y and it does not depend on x, as shown in Figure 6.

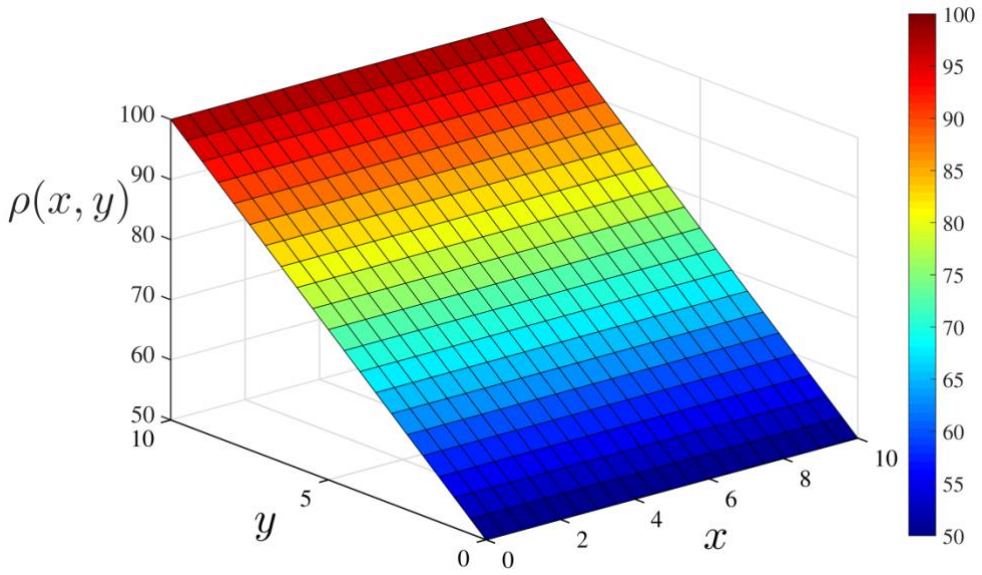

Figure 6: Numerical solution for $\bar{j}_{\text{left}} = \bar{j}_{\text{right}} = 0$, $\bar{\rho}_{\text{upper}} = 100\ \text{mol}\cdot\text{m}^{-3}$ and $\bar{\rho}_{\text{lower}} = 50\ \text{mol}\cdot\text{m}^{-3}$ obtained using a 21×21 grid.

3) For a specified discretization, the values of $\bar{\rho}_{\text{upper}}$, $\bar{\rho}_{\text{lower}}$ $\bar{j}_{\text{left}} / D$, $\bar{j}_{\text{right}} / D$ do not affect the coefficient matrix A but only the vector b. Accordingly, A is created ***once*** while b is reconstructed for different pairs $\{\bar{\rho}_{\text{upper}}, \bar{\rho}_{\text{lower}}\}$.

Since for $\bar{j}_{\text{left}} = \bar{j}_{\text{right}} = 0$, the solutions depend linearly on y and they do not depend on x, a linear plot of the numerical solutions at $x = L_x / 2$ is provided in Figure 7 for different pairs $\{\bar{\rho}_{\text{upper}}, \bar{\rho}_{\text{lower}}\}$.

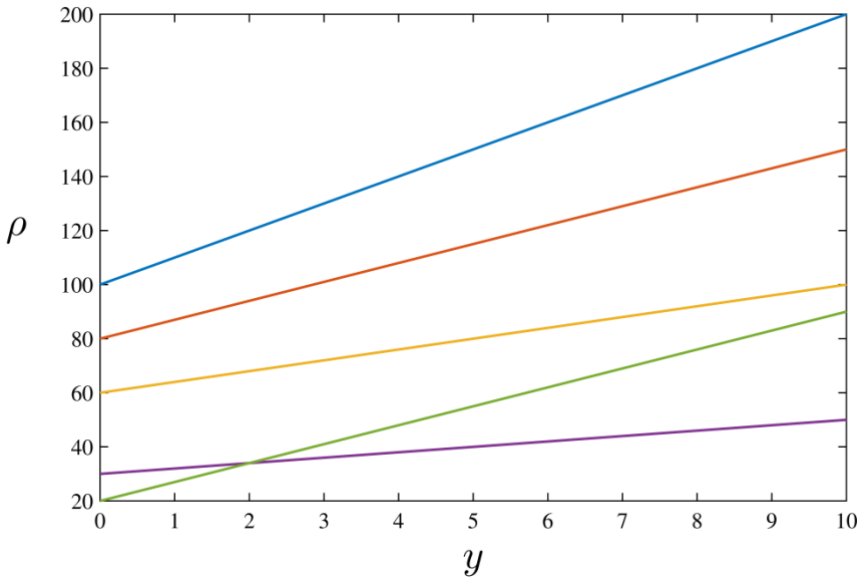

Figure 7: Numerical solutions $\rho(x, y)$ vs. y, for $\bar{j}_{\text{left}} = \bar{j}_{\text{right}} = 0$ and for five different pairs $\{\bar{\rho}_{\text{upper}}, \bar{\rho}_{\text{lower}}\} = \{200,100\},\{150,80\},\{100,60\},\{50,30\},\{90,20\}$. All the results have been obtained using a 21×21 grid.

4) The analytical (exact) solution for $\bar{j}_{\text{left}} = \bar{j}_{\text{right}} = 0$ reads

$$\rho(x, y) = \bar{\rho}_{\text{lower}} + \frac{\bar{\rho}_{\text{upper}} - \bar{\rho}_{\text{lower}}}{L_y} y \,. \tag{5.1}$$

Because of this linearity, the approximations defined by Eqs. (2.1)-(2.3) are ***satisfied exactly*** (they convert to equalities). Therefore the error between the analytical and the numerical solution (even of a coarse grid) will be (almost) zero everywhere (up to computational round-off errors due to floating point arithmetic). For example, for a 5×5 grid the maximum error is $\varepsilon_{\max} = 4.2633 \cdot 10^{-14}$ which close to the machine precision. Accordingly, the boundary conditions $\bar{j}_{\text{left}} = \bar{j}_{\text{right}} = 0$ and $\bar{\rho}_{\text{upper}}$, $\bar{\rho}_{\text{lower}}$ uniform are not suitable to study discretization errors of the finite difference method used.

An alternative option is to consider non-zero $\bar{j}_{\text{left}}$ and/or $\bar{j}_{\text{right}}$. However, the analytical solution in this case involves an infinite series:

$$\rho(x, y) = \bar{\rho}_{\text{lower}} + \frac{\bar{\rho}_{\text{upper}} - \bar{\rho}_{\text{lower}}}{L_y} y + \sum_{n=0}^{\infty} c_n(x) \sin\left(\frac{(2n+1)\pi}{L_y} y\right) \tag{5.2}$$

where

$$c_n(x) = 4L_y \frac{\bar{j}_{\text{right}} \cosh\left((2n+1)\pi x / L_y\right) - \bar{j}_{\text{left}} \cosh\left((2n+1)\pi (L_x - x) / L_y\right)}{(2n+1)^2 \pi^2 \sinh\left((2n+1)\pi L_x / L_y\right)}, \tag{5.3}$$

and therefore the computation of the exact solution inserts an additional truncation error by the necessity of using finite number of terms from the series.

The pointwise error of the numerical solution at the node (i, j) is defined as $\varepsilon(x_i, y_j) := \left|\rho_{i,j} - \rho(x_i, y_j)\right|$, where $\rho_{i,j}$ is the computed value of the numerical solution at the point (x_i, y_j), and $\rho(x_i, y_j)$ the corresponding exact value. In Figures 9, 10 the pointwise error is plotted for a coarse grid 5×5 and a finer grid 100×100, respectively.

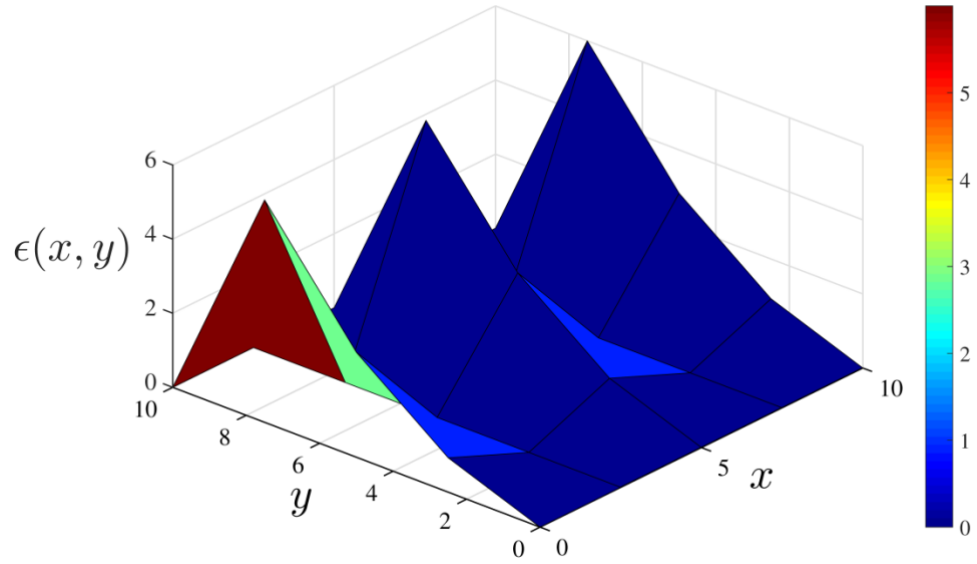

Figure 8: Pointwise error of the numerical solution for 5×5 discretization. Maximum error is $\varepsilon_{\max}$=5.99936.

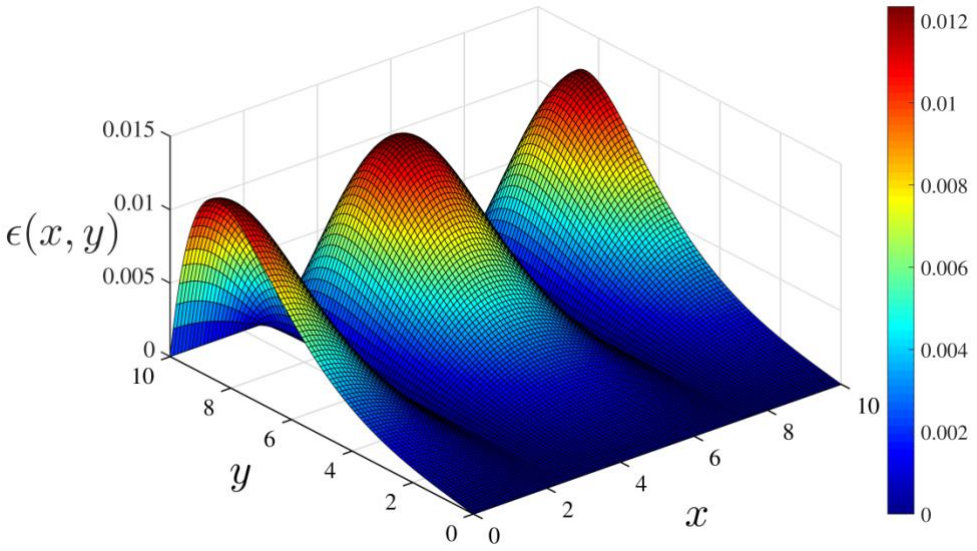

Figure 9: Pointwise error of the numerical solution for 100×100 discretization. Maximum error is $\varepsilon_{\max}$=0.0123398.

In both grids the biggest discrepancy between the numerical and the exact solution appear close to the top boundary where the variation of the exact solution is large. In fact, in the central finite difference scheme we have used, the approximations of first derivatives (see Eq. (2.1)) are exact for polynomial functions of degree less than or equal to 2, while the approximations of second derivatives are exact for polynomial functions of degree less than or equal to 3. Accordingly, the pointwise error is bigger in regions where higher-order derivatives of the exact solution are rather large.

5) Increasing the number of nodes decreases the error. To verify this numerically, the maximum error $\varepsilon_{\max} := \max_{i,j} \varepsilon(x_i, y_j)$ is computed for different number of nodes. In particular, different $N \times N$ grids are employed and $\varepsilon_{\max}$ is plotted as a function of the nodal distance $h := h_x = h_y$ (which is inversely proportional to $N-1$) in Figure 10. Consequently, the slope of the straight-like line in Figure 11 gives the order p.

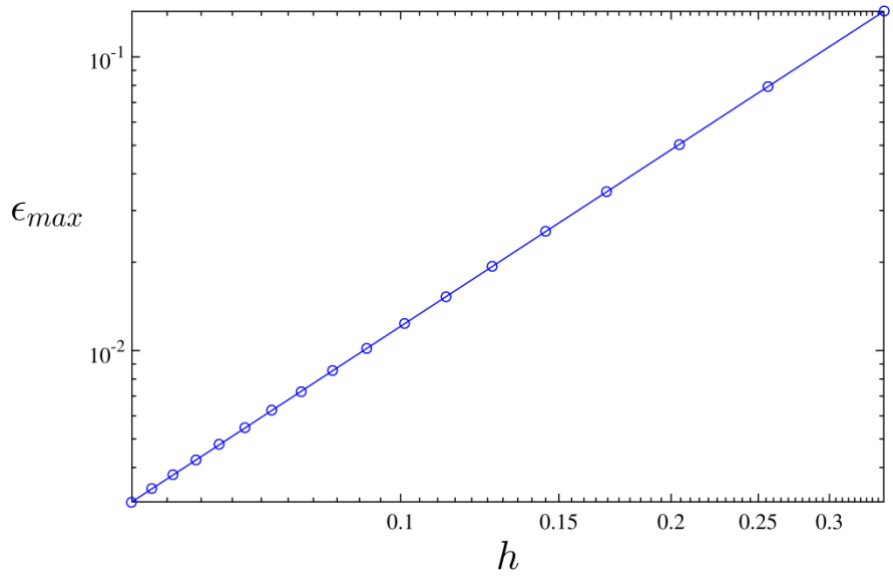

Figure 10: Maximum pointwise error as a function of the nodal distance, using different $N \times N$ grids with $N = 30:10:200$.

6) When $\bar{j}_{\text{left}} = \bar{j}_{\text{right}} = 0$ and $\bar{\rho}_{\text{upper}}$, $\bar{\rho}_{\text{lower}}$ are arbitrary constants (i.e. they don't depend on x), the exact solutions depend linearly on y and they do not depend on x (see Eq. (5.1)). Therefore the Laplace equation reduces to $\partial_{yy}\rho(y) = 0$ and the simulations can be speeded up by solving the 1-D boundary value problem

$$\begin{cases} \rho''(y) = 0, \quad 0 < y < L_y \\ \rho(0) = \bar{\rho}_{\text{lower}}, \quad \rho(L_y) = \bar{\rho}_{\text{upper}} \end{cases} . \tag{5.4}$$

The finite differences counterpart of this problem is

$$\frac{\rho_{j+1}-2\rho_j+\rho_{j-1}}{h_y^2}=0, \qquad j=2,\ldots,(N_y-1)$$
$$\rho_1=\bar{\rho}_{\text{lower}}, \quad \rho_{N_y}=\bar{\rho}_{\text{upper}} \tag{5.5}$$

Where the numbering of nodes begins at $y=0$ ($j=1$) and ends at $y=L_y$ ($j=N_y$). Eq. (5.13)$_1$ can also be written as $-\rho_{j+1}+2\rho_j-\rho_{j-1}=0$ and in turn the problem converts to a $(N_y-2)\times(N_y-2)$ system of linear equations $Ax=b$ as

$$\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix} \begin{bmatrix} \rho_2 \\ \rho_3 \\ \vdots \\ \rho_{N_y-2} \\ \rho_{N_y-1} \end{bmatrix} = \begin{bmatrix} \bar{\rho}_{\text{lower}} \\ 0 \\ \vdots \\ 0 \\ \bar{\rho}_{\text{upper}} \end{bmatrix}. \tag{5.6}$$

Then, the solution of this system can be replicated N_x times to obtain the solution of the corresponding 2-D problem (see the respective Matlab code in Appendix). The results will be identical to those presented in Figures 5-7.

7) In Figures 12-17, the numerical results for different ***uniform*** boundary values $\bar{\rho}_{\text{upper}}$, $\bar{\rho}_{\text{lower}}$, $\bar{j}_{\text{left}}$, $\bar{j}_{\text{right}}$ with ***non-zero*** $\bar{j}_{\text{left}}$ and/or $\bar{j}_{\text{right}}$ are shown. In this case, the analytical solution is nonlinear and it has an infinite series form presented in Eq. (5.2). To obtain an acceptable approximate solution by the present finite difference scheme, a discretization with 300×300 nodes has been adopted in all these simulations.

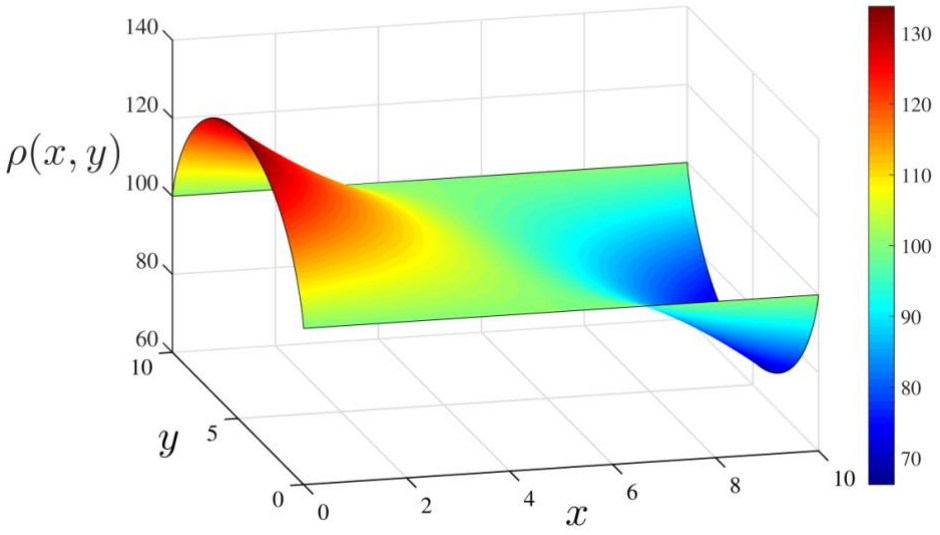

Figure 12: Numerical solution for $\bar{\rho}_{\text{upper}}=\bar{\rho}_{\text{lower}}=100\ \text{mol}\cdot\text{m}^{-3}$, $\bar{j}_{\text{left}}/D=-10\ \text{mol}\cdot\text{m}^{-4}$, and $\bar{j}_{\text{right}}/D=10\ \text{mol}\cdot\text{m}^{-4}$.

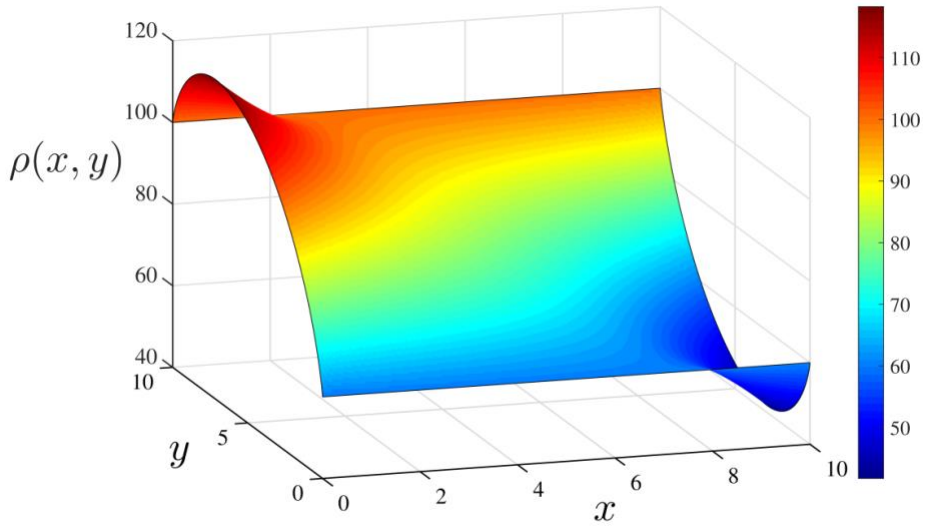

Figure 13: Numerical solution for $\bar{\rho}_{\text{upper}} = 100\ \text{mol}\cdot\text{m}^{-3}$, $\bar{\rho}_{\text{lower}} = 60\ \text{mol}\cdot\text{m}^{-3}$, $\bar{j}_{\text{left}} / D = -10\ \text{mol}\cdot\text{m}^{-4}$, and $\bar{j}_{\text{right}} / D = 10\ \text{mol}\cdot\text{m}^{-4}$.

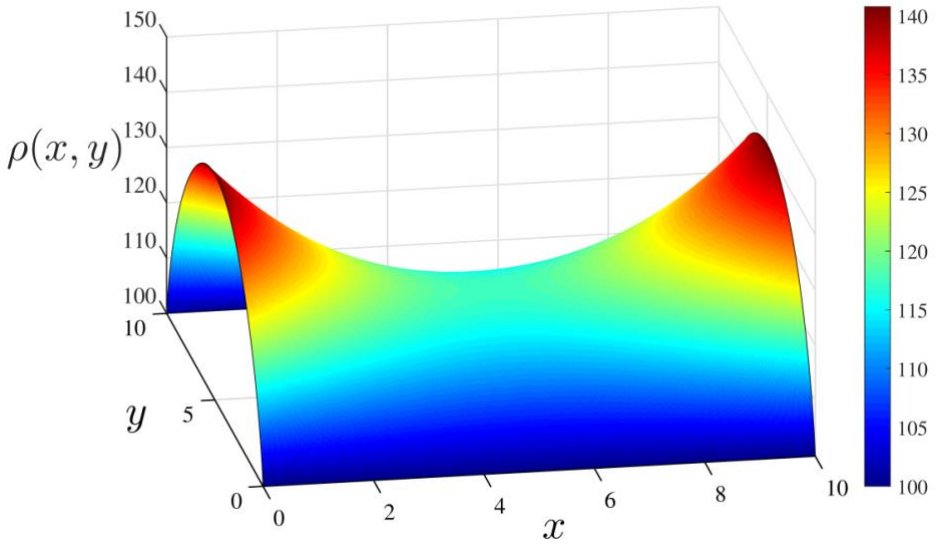

Figure 14: Numerical solution for $\bar{\rho}_{\text{upper}} = \bar{\rho}_{\text{lower}} = 100\ \text{mol}\cdot\text{m}^{-3}$, $\bar{j}_{\text{left}} / D = \bar{j}_{\text{right}} / D = -10\ \text{mol}\cdot\text{m}^{-4}$.

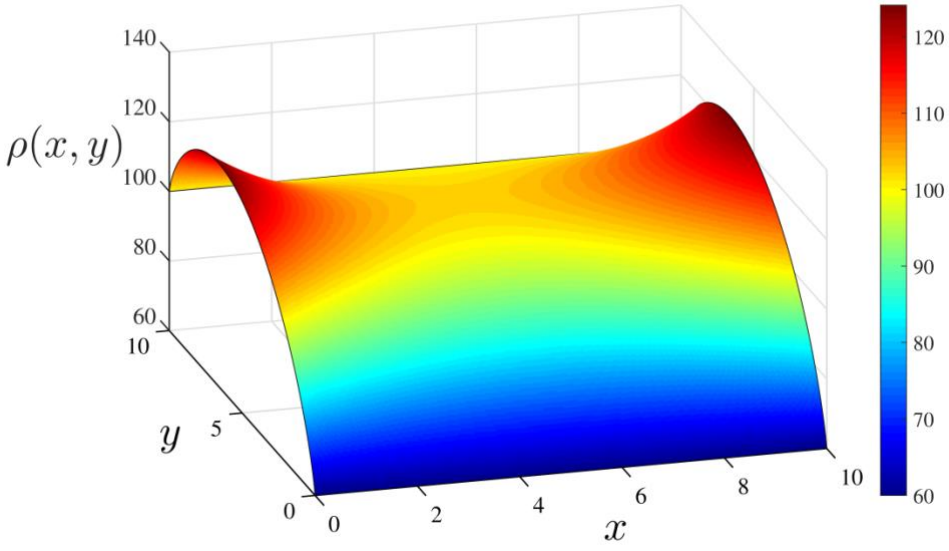

Figure 15: Numerical solution for $\bar{\rho}_{\text{upper}} = 100\ \text{mol}\cdot\text{m}^{-3}$, $\bar{\rho}_{\text{lower}} = 60\ \text{mol}\cdot\text{m}^{-3}$, $\bar{j}_{\text{left}} / D = \bar{j}_{\text{right}} / D = -10\ \text{mol}\cdot\text{m}^{-4}$.

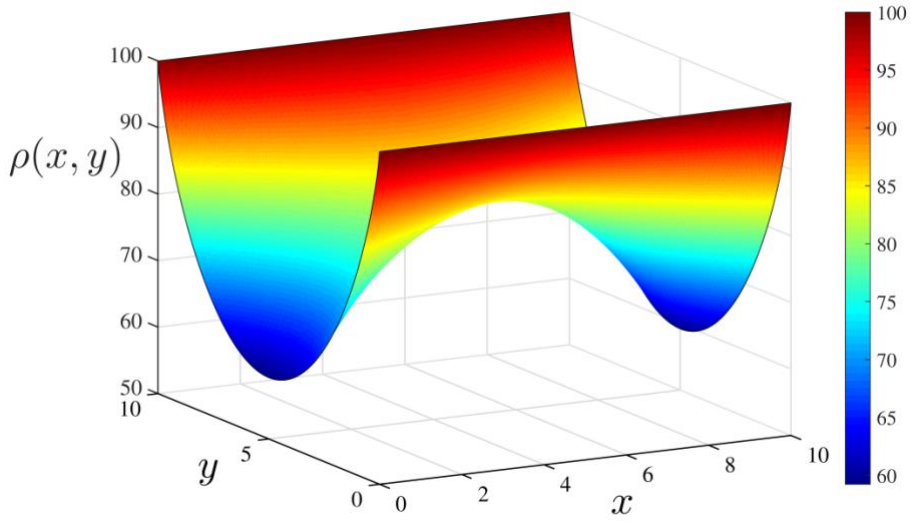

Figure 16: Numerical solution for $\bar{\rho}_{\text{upper}} = \bar{\rho}_{\text{lower}} = 100\ \text{mol}\cdot\text{m}^{-3}$, $\bar{j}_{\text{left}} / D = \bar{j}_{\text{right}} / D = 10\ \text{mol}\cdot\text{m}^{-4}$.

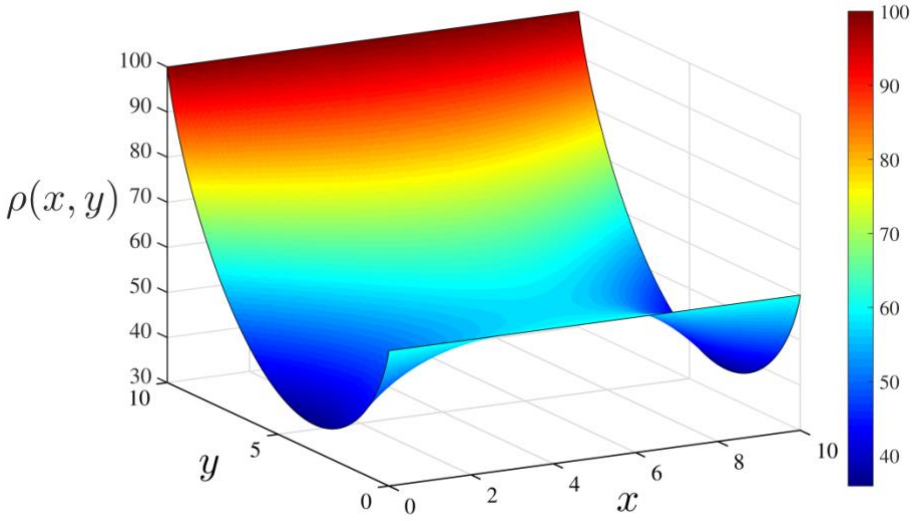

Figure 17: Numerical solution for $\bar{\rho}_{\text{upper}} = 100\ \text{mol}\cdot\text{m}^{-3}$, $\bar{\rho}_{\text{lower}} = 60\ \text{mol}\cdot\text{m}^{-3}$, $\bar{j}_{\text{left}} / D = \bar{j}_{\text{right}} / D = 10\ \text{mol}\cdot\text{m}^{-4}$.

It is noted that the following six cases are illustrated in Figures 12-17:

i) $\bar{\rho}_{\text{upper}} = \bar{\rho}_{\text{lower}}$, $\bar{j}_{\text{left}} < 0$, $\bar{j}_{\text{right}} > 0$, i.e. equal concentrations at top and lower boundaries, inlet flux at the left boundary and outlet flux at right boundary (Figure 12).

ii) $\bar{\rho}_{\text{upper}} \neq \bar{\rho}_{\text{lower}}$, $\bar{j}_{\text{left}} < 0$, $\bar{j}_{\text{right}} > 0$, i.e. different concentrations at top and lower boundaries, and left/right fluxes as in (i) (Figure 13).

iii) $\bar{\rho}_{\text{upper}} = \bar{\rho}_{\text{lower}}$, $\bar{j}_{\text{left}} < 0$, $\bar{j}_{\text{right}} < 0$, i.e. equal concentrations at top and lower boundaries, and inlet fluxes at both left and right boundary (Figure 14).

iv) $\bar{\rho}_{\text{upper}} \neq \bar{\rho}_{\text{lower}}$, $\bar{j}_{\text{left}} < 0$, $\bar{j}_{\text{right}} < 0$, i.e. different concentrations at top and lower boundaries, and inlet fluxes at both left and right boundary (Figure 15).

v) $\bar{\rho}_{\text{upper}} = \bar{\rho}_{\text{lower}}$, $\bar{j}_{\text{left}} > 0$, $\bar{j}_{\text{right}} > 0$, i.e. equal concentrations at top and lower boundaries, and outlet fluxes at both left and right boundary (Figure 16).

vi) $\bar{\rho}_{\text{upper}} \neq \bar{\rho}_{\text{lower}}$, $\bar{j}_{\text{left}} > 0$, $\bar{j}_{\text{right}} > 0$, i.e. different concentrations at top and lower boundaries, and outlet fluxes at both left and right boundary (Figure 17).

References

Holmes, M. H. (2007): *Introduction to numerical methods in differential equations* (Vol. 52). Springer, New York.

Iserles, A. (2009): *A first course in the numerical analysis of differential equations* (2nd ed.). Cambridge University Press, New York.

Knabner, P., and Angermann, L. (2003): *Numerical methods for elliptic and parabolic partial differential equations* (Vol. 44). Springer-Verlag, New York.

Strikwerda, J. C. (2004): *Finite difference schemes and partial differential equations* (2nd ed.). Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA.

Wikipedia, the free encyclopedia: *Fick's laws of diffusion*.

APPENDIX

Table A.1: Implementation of the present finite difference scheme in Matlab

```
function [x,y,R]=FinDiff_Laplace_Eq(Lx,Ly,Nx,Ny,r_upper,r_lower,j_left,j_right,flag)

% Finite Difference Method for the numerical solution of
% 2D Laplace equation in a rectangular domain of width Lx and height Ly
% Nx and  Ny are the numbers of nodes in x and y directions, respectively
% Dirchlet BCs are used on the upper and lower side of the domain
% and Neumann BCs are used on the left and right side of the domain
% In particular:
% r_upper = prescribed concentration at the upper side
% r_lower = prescribed concentration at the lower side
% j_left = prescribed flux (normalized by D) at left side
% j_right = prescribed flux (normalized by D) at right side
% flag = 1 plots the result

% NOTE: r_upper,r_lower,j_left,j_right can be scalar or
% row vectors (for non-uniform BCs)

hx = Lx/(Nx-1); % distance of nodes in x direction
hy = Ly/(Ny-1); % distance of nodes in y direction
x = [0:Nx-1]*hx; % x-coordinates of nodes
y = [0:Ny-1]*hy; % y-coordinates of nodes

% Construct the coefficient matrix A of the final linear system Ax = b
A = buildCoeffMatrix(Lx,Ly,Nx,Ny);

% Construct the vector b of the final linear system Ax = b
b = buildRHSvector(Lx,Ly,Nx,Ny,r_upper,r_lower,j_left,j_right);

% Solve the system Ax=b
r = A\b;
% Insert the known values for the Dirichlet BCs in the solution vector:
r =[r_upper'.*ones(Nx,1); r; r_lower'.*ones(Nx,1)];

% Reshape the column r into a matrix R such that R(j,i) = ρ(x(i),y(j))
% to be used for surface plots
R = reshape(r,[Ny,Nx])'; R=flip(R);


%% ---------------------------------------------------------------------
% Surface plot of the solution ρ(x,y)
if flag==1,
    fig1=figure(1); fig1.Position = [10    45   985   640];

    hh=surf(x,y,R);
    colormap('jet'); colorbar;
    % don't show mesh lines for fine grids but draw boundary of the surface
        if Nx*Ny>5000
             set(hh,'linestyle','none');   hold on;
             plot3(zeros(Nx,1),y,R(:,1),'k'); % line on left boundary
             plot3(Lx*ones(Nx,1),y,R(:,end),'k'); % line on right boundary
             plot3(x,zeros(Ny,1),R(1,:),'k'); % line on lower boundary
             plot3(x,Ly*ones(Ny,1),R(end,:),'k'); % line on top boundary
        end

    set(gca,'FontSize',18,'FontName','Times');
    xlabel('$$x$$','Interpreter','Latex','FontSize',33);
    ylabel('$$y$$','Interpreter','Latex','FontSize',33,'Rotation',0);
    zlabel('$$\rho(x,y)$$','Interpreter','Latex','FontSize',33,'Rotation',0)
    set(gca, 'LineWidth', 1,'XColor','k','YColor','k'); set(gcf,'Color','w');
end
```

Table A.2: Function which constructs the coefficient matrix A in the final linear system $Ax = b$

```
function A = buildCoeffMatrix(Lx,Ly,Nx,Ny)

hx = Lx/(Nx-1); % distance of nodes in x direction
hy = Ly/(Ny-1); % distance of nodes in y direction

% numbers of bourdary nodes of dirichlet and neumann BCs
dirichlet = [1:Nx, (Ny-1)*Nx+1:Nx*Ny];
left=(1:(Ny-2))*Nx+1;
right=(2:(Ny-1))*Nx;

% Construct the coefficient matrix A of the final linear system Ax = b
% First we constract A without taking into account BCs
n=Ny*Nx; % initial shape without taking into account BCs is nxn
e = ones(n,1);
ax=(1/hx^2); ay=(1/hy^2); axy=-2*(ax + ay); % non-zero values in A
A = spdiags([ay*e,ax*e,axy*e,ax*e,ay*e],[-Nx, -1:1, Nx],n,n);

% Taking into account Neumann BCs:
A(left,left+1)=2*A(left,left+1);      A(left,left-1)=0;
A(right,right-1)=2*A(right,right-1);  A(right,right+1)=0;
% delete rows and columns corresponding to Dirichlet BCs:
A(dirichlet,:)=[];  A(:,dirichlet)=[];
```

Table A.3: Function which constructs the right hand side vector b in the final linear system $Ax = b$

```
function b = buildRHSvector(Lx,Ly,Nx,Ny,r_upper,r_lower,j_left,j_right)
hx = Lx/(Nx-1); % distance of nodes in x direction
hy = Ly/(Ny-1); % distance of nodes in y direction

% numbers of bourdary nodes of dirichlet and neumann BCs
dirichlet = [1:Nx, (Ny-1)*Nx+1:Nx*Ny];
left=(1:(Ny-2))*Nx+1;
right=(2:(Ny-1))*Nx;

n=Ny*Nx;
b1 = sparse(n,1); % an empty sparse vector
b1(left)=2*j_left/hx;    b1(right)=2*j_right/hx; % Neumann BCs
b1(dirichlet)=[];  % delete rows corresponding to Dirichlet BCs
b2 = sparse((Ny-2)*Nx,1); % an empty sparse vector
b2(1:Nx)=r_upper/hy^2;   b2(end-Nx+1:end)=r_lower/hy^2;
b = b1-b2;
```

Table A.4: Main program used in the test problems 1, 2 and 7 of Task 5.2

```
clc; close all; clear all
Lx=10;   Ly=10;  %width Lx and height Ly of the domain
Nx=300;  Ny=300; % number of nodes in x and y directions

% Boundary Conditions (BCs)
r_upper=100;
r_lower=60;
j_left=10;   % flux normalized by diffusion coefficient D
j_right=10;  % flux normalized by diffusion coefficient D

[x,y,R]=FinDiff_Laplace_Eq(Lx,Ly,Nx,Ny,r_upper,r_lower,j_left,j_right,1);
```

Table A.5: Main program used in the test problem 3 of Task 5.2

```
clc; close all; clear all
Lx=10;   Ly=10;  %width Lx and height Ly of the domain
Nx=21;  Ny=21; % number of nodes in x and y directions

A = buildCoeffMatrix(Lx,Ly,Nx,Ny); % Build Coefficient Matrix

% Boundary Conditions (BCs)
j_left=0;   % flux normalized by diffusion coefficient D
j_right=0;  % flux normalized by diffusion coefficient D
% Different pairs of Dirichlet BCs:
r_upper=[200, 150, 100, 50, 90];
r_lower=[100, 80, 60, 30, 20];

hy = Ly/(Ny-1);   y = [0:Ny-1]*hy;

fig2=figure(2); fig2.Position = [10    45   985   640];
for i = 1:length(r_upper)
    % Build right hand side vector b
    b = buildRHSvector(Lx,Ly,Nx,Ny,r_upper(i),r_lower(i),j_left,j_right);
    % Solve the system Ax=b
    r = A\b;
    % Insert the known values for the Dirichlet BCs in the solution vector:
    r =[r_upper(i)*ones(Nx,1); r; r_lower(i)*ones(Nx,1)];
    % Reshape the column r into a matrix R
    R = reshape(r,[Ny,Nx])'; R=flip(R);
    plot(y,R(:,round((Nx+1)/2)),'Linewidth',2);
    hold all
end

set(gca,'FontSize',18,'FontName','Times');
xlabel('$$y$$','Interpreter','Latex','FontSize',33);
ylabl = ylabel('$$\rho$$','Interpreter','Latex','FontSize',33,'Rotation',0);
set(ylabl, 'Units', 'Normalized', 'Position', [-0.1, 0.45, 0]);
set(gca, 'LineWidth', 1,'XColor','k','YColor','k'); set(gcf,'Color','w');
```

Table A.6: Main program used in the test problem 4 of Task 5.2

```
clc; close all; clear all
Lx=10;   Ly=10;  %width Lx and height Ly of the domain
Nx=100;  Ny=100; % number of nodes in x and y directions

hx = Lx/(Nx-1); % distance of nodes in x direction
x = [0:Nx-1]*hx; % x-coordinates of nodes

% Boundary Conditions (BCs)
r0=100; % Constant to be used in r_upper(x)
r_upper= r0 + r0*cos((2*pi*x)/Lx);
% r_upper=100;
r_lower=50;
j_left=0;   % flux normalized by diffusion coefficient D
j_right=0;  % flux normalized by diffusion coefficient D

% numerical solution
[x,y,R]=FinDiff_Laplace_Eq(Lx,Ly,Nx,Ny,r_upper,r_lower,j_left,j_right,0);

% Exact solution
% R_exact=@(x,y)  r_lower + (r_upper-r_lower)*y/Ly;
R_exact=@(x,y) r_lower + ((r0 - r_lower)*y)/Ly + ...
    r0*( sinh((2*pi*y)/Lx)/sinh((2*pi*Ly)/Lx) ).*cos((2*pi*x)/Lx);
[X,Y] = meshgrid(x,y);
Rexact = R_exact(X,Y);

% error calculation
error = abs(R - Rexact);
fprintf('maximum error is %g \n',max(max(error)))
```

```
%% ------------------------------------------------------------
fig1=figure(1); fig1.Position = [10    45   985   640];
surf(x,y,error); colormap('jet'); colorbar
set(gca,'FontSize',18,'FontName','Times');
xlabel('$$x$$','Interpreter','Latex','FontSize',33);
ylabel('$$y$$','Interpreter','Latex','FontSize',33,'Rotation',0);
zlabel('$$\epsilon(x,y)$$','Interpreter','Latex','FontSize',33,'Rotation',0)
set(gca, 'LineWidth', 1,'XColor','k','YColor','k'); set(gcf,'Color','w');

%-----------------------------------------------------------
% Plot the exact solution using a fine (x,y) mesh
x=linspace(0,Lx,101); y=linspace(0,Ly,101);
[X,Y] = meshgrid(x,y);
fig2=figure(2); fig2.Position = [10    45   985   640];
hh=surf(X,Y,R_exact(X,Y));
colormap('jet'); shading interp; colorbar
set(gca,'FontSize',18,'FontName','Times');
xlabel('$$x$$','Interpreter','Latex','FontSize',33);
ylabel('$$y$$','Interpreter','Latex','FontSize',33,'Rotation',0);
zlabel('$$\rho(x,y)$$','Interpreter','Latex','FontSize',33,'Rotation',0)
set(gca, 'LineWidth', 1,'XColor','k','YColor','k'); set(gcf,'Color','w');
```

Table A.7: Main program used in the test problem 5 of Task 5.2

```
clc; close all; clear all
Lx=10;   Ly=10;  %width Lx and height Ly of the domain

% Boundary Conditions (BCs)
r0=100; % Constant to be used in r_upper(x)
r_lower=50;
j_left=0;   % flux normalized by diffusion coefficient D
j_right=0;  % flux normalized by diffusion coefficient D

% Exact solution
R_exact=@(x,y) r_lower + ((r0 - r_lower)*y)/Ly + ...
    r0*( sinh((2*pi*y)/Lx)/sinh((2*pi*Ly)/Lx) ).*cos((2*pi*x)/Lx);

N = 30:10:200;   % different values for Nx=Ny
error_vals = zeros(size(N)); % vector to put max error for different N
h_vals = zeros(size(N)); % vector to put h=hx=hy  values for different N

for i=1:length(N)

    Nx=N(i);  Ny=Nx; % number of nodes in x and y directions

    hx = Lx/(Nx-1); % distance of nodes in x direction
    x = [0:Nx-1]*hx; % x-coordinates of nodes
    % Boundary Condition at top boundary
    r_upper= r0 + r0*cos((2*pi*x)/Lx);

    % numerical solution
    [x,y,R]=FinDiff_Laplace_Eq(Lx,Ly,Nx,Ny,r_upper,r_lower,j_left,j_right,0);

    % calculate exact solution at nodes
    [X,Y] = meshgrid(x,y);
    Rexact = R_exact(X,Y);

    % error calculation:
    error = abs(R - Rexact); % pointwise error
    error_vals(i)=max(max(error)); % max value of error
    fprintf('N=%d, maximum error is %g \n', Nx,error_vals(i))
    h_vals(i)=hx;
end

% numerical estimation of the order of the method:
for i=1:length(N)-1
    p = log(error_vals(i+1)/error_vals(i))/log(h_vals(i+1)/h_vals(i));
    fprintf('N1 = %d, N2=%d:  order p= %g \n', N(i),N(i+1),p)
```

```
end

% Logarithmic plot of error_max vs h=hx=hy
fig1=figure(1); fig1.Position = [10    45   985   640];
loglog(h_vals,error_vals,'b-o','LineWidth',1,'MarkerSize',8)
axis tight
set(gca,'FontSize',18,'FontName','Times');
xlabel('$$h$$','Interpreter','Latex','FontSize',33);
ylabel('$$\epsilon_{max}$$','Interpreter','Latex','FontSize',33,'Rotation',0);
set(gca, 'LineWidth', 1,'XColor','k','YColor','k'); set(gcf,'Color','w');
```

Table A.8: Main program used in the test problem 6 of Task 5.2

```
clc; close all; clear all
% Solution as 1-D problem in the case of j_left=0, j_right=0;
% and uniform r_upper and r_lower

Lx=10;   Ly=10;  %width Lx and height Ly of the domain
Nx=21;  Ny=21; % number of nodes in x and y directions

% Boundary Conditions (BCs)
r_upper=100;
r_lower=50;
j_left=0;   % flux normalized by diffusion coefficient D
j_right=0;  % flux normalized by diffusion coefficient D

hx = Lx/(Nx-1); % distance of nodes in x direction
hy = Ly/(Ny-1); % distance of nodes in y direction
x = [0:Nx-1]*hx; % x-coordinates of nodes
y = [0:Ny-1]*hy; % y-coordinates of nodes

% Construct the coefficient matrix A and vector b of the linear system Ax = b
e = ones(Ny-1,1);
A = spdiags([-e,2*e,-e],-1:1,Ny-2,Ny-2);
b = sparse(Ny-2,1); b(1)=r_lower;  b(end)=r_upper;
% Solve the system Ax=b
r = A\b;
% Insert the known values for the Dirichlet BCs in the solution vector:
r =[r_lower; r; r_upper];

% replicate the solution Nx times
R=repmat(r,1,Nx);

%%
%Surface plot of the solution ρ(x,y)
fig1=figure(1); fig1.Position = [10    45   985   640];
hh=surf(x,y,R); % for 3D plot
% hh=surface(x,y,R); %set(hh,'linestyle','none') % for 2D plot
colormap('jet'); colorbar; %set(hh,'linestyle','none')

set(gca,'FontSize',18,'FontName','Times');
xlabel('$$x$$','Interpreter','Latex','FontSize',33);
ylabel('$$y$$','Interpreter','Latex','FontSize',33,'Rotation',0);
zlabel('$$\rho(x,y)$$','Interpreter','Latex','FontSize',33,'Rotation',0)
set(gca, 'LineWidth', 1,'XColor','k','YColor','k'); set(gcf,'Color','w');
```