Assignment 2 — Poisson's equation in a $\mathbb{R}^2$ annulus

2019862s

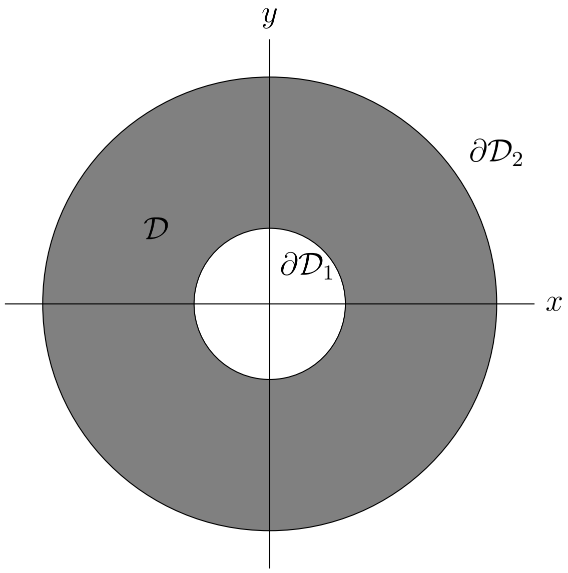
Semester 1, 2015–2016

The purpose of this assignment is to numerically solve Poisson's equation using the Gauss-Jacobi iterative method. Consider the following problem:

$$\begin{aligned} \nabla^2 u(x,y) &= -2, && (x,y) \in \mathcal{D}, \\ u(x,y) &= \sin\big(\tan^{-1}(y/x)\big), && \text{on} \quad (x,y) \in \partial\mathcal{D}_1, \\ u(x,y) &= \cos\big(3\tan^{-1}(y/x)\big), && \text{on} \quad (x,y) \in \partial\mathcal{D}_2, \end{aligned} \tag{1}$$

where $\mathcal{D} \subset \mathbb{R}^2$ is the closed annular region between the circle $\partial\mathcal{D}_1$ of radius 1 and the circle $\partial\mathcal{D}_2$ of radius 3, both centered at $(0,0)$ as shown in Figure 1.

Figure 1: The annular region $\mathcal{D}$.



For the ease of the reader, each question is explicitly stated before presenting its solution. Further, any code used and/or graph produced is listed as well.

Question 1. *The transformations from Cartesian coordinates* (x, y) *to polar coordinates* (r, θ) *are given by*

$$r = \sqrt{x^2 + y^2}, \qquad \theta = \tan^{-1}(y/x),$$

where in polar coordinates the Laplacian operator takes the form

$$\nabla^2 = \frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + \frac{1}{r^2}\frac{\partial^2}{\partial \theta^2}.$$

Using these facts, transform problem (1), including the PDE, the boundary conditions and the domain of integration, from (x, y) *to* (r, θ)*. You should find that the domain of integration is a rectangular region in terms of* r *and* θ*.*

Solution. Using the given transformations, the PDE becomes

$$\nabla^2 u(x, y) = -2 \quad \longrightarrow \quad \nabla^2 u(r, \theta) = \frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} + \frac{1}{r^2}\frac{\partial^2 u}{\partial \theta^2} = -2,$$

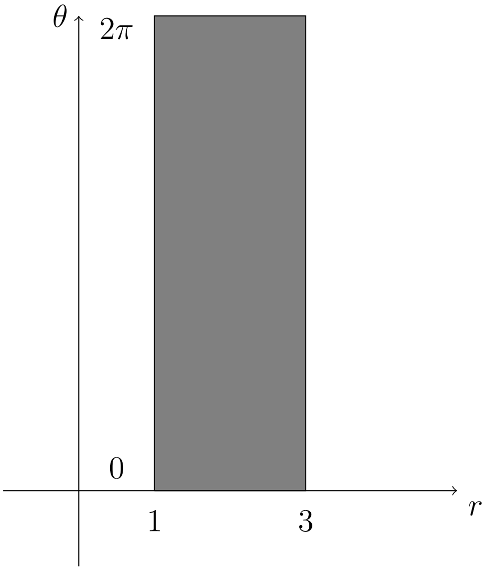
$$\therefore \qquad u_{rr} + \frac{1}{r}u_r + \frac{1}{r^2}u_{\theta\theta} = -2, \qquad r \in [1, 3], \quad \theta \in [0, 2\pi].$$

Transforming the boundary conditions yields

$$\begin{aligned} u(x, y) &= \sin\left(\tan^{-1}(y/x)\right) \quad \text{on} \quad (x, y) \in \partial\mathcal{D}_1 \quad \longrightarrow \quad u(r, \theta) = \sin(\theta), \qquad r = 1, \quad \theta \in [0, 2\pi], \\ u(x, y) &= \cos\left(3\tan^{-1}(y/x)\right) \quad \text{on} \quad (x, y) \in \partial\mathcal{D}_2 \quad \longrightarrow \quad u(r, \theta) = \cos(3\theta), \qquad r = 3, \quad \theta \in [0, 2\pi]. \end{aligned}$$

This forms the rectangular region shown in Figure 2, as required.

Figure 2: The rectangular region in polar coordinates.



□

Question 2. *Set up a rectangular grid, composed of $n+1$ nodes in the radial direction and $m+1$ nodes in the azimuthal direction. Define grid spacings Δr and $\Delta\theta$ and grid points r_i and θ_j. Introduce appropriate notation to describe the approximation to the solution of BVP (1) at each node. Provide a sketch of the grid. Remember to list the range of values for the indices i and j where appropriate.* ***Use the* MATLAB *convention that indices MUST start from 1, rather than the usual mathematical convention where indices start from 0.***

Solution. The grid spacing in the radial and azimuthal directions are defined respectively as follows:

$$\Delta r = \frac{2}{n}, \qquad \Delta\theta = \frac{2\pi}{m}.$$

Hence, the expressions for the i-th radius, and the j-th angle using the MATLAB convention are given by

$$r_i = r_1 + \big(i-1\big)\Delta r = \frac{n-2}{n} + i\frac{2}{n}, \quad \text{for} \quad i \in [1, n+1],$$

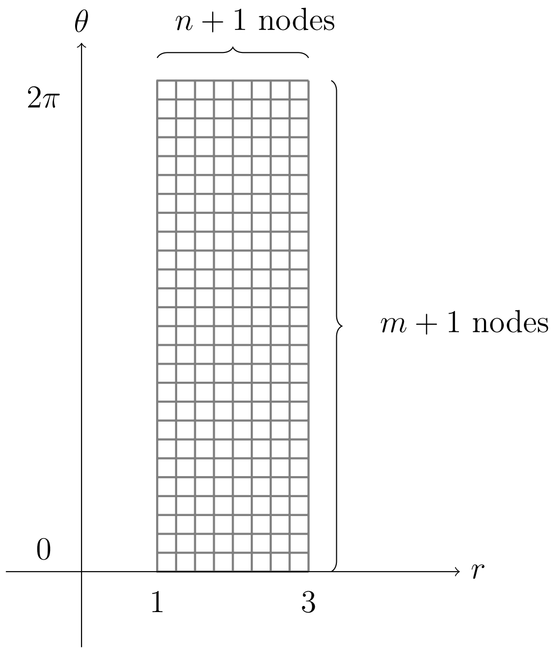
$$\theta_j = \theta_1 + \big(j-1\big)\Delta\theta = \frac{-2\pi}{m} + j\frac{2\pi}{m}, \quad \text{for} \quad j \in [1, m+1].$$

And the solution to the BVP at the i,j-th node is

$$u(r_i, \theta_j) = u_{i,j}.$$

Figure 3 provides a sketch of the grid.

Figure 3: The rectangular region in polar coordinates.



$\square$

Question 3. *Use second-order of convergence, centered-difference approximations to the derivatives to generate difference equations that correspond to the PDE discretised at the interior nodes. Explicitly state the ranges of the i and j indices.*

Solution. Using the appropriate approximations, each term of the PDE discretised at the interior nodes becomes

$$u_{rr} = \left(\frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{(\Delta r)^2}\right) + O((\Delta r)^2),$$

$$u_r = \left(\frac{u_{i+1,j} - u_{i-1,j}}{2\Delta r}\right) + O((\Delta r)^2),$$

$$u_{\theta\theta} = \left(\frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{(\Delta\theta)^2}\right) + O((\Delta\theta)^2).$$

Hence, the PDE at the interior nodes is given by

$$\left(\frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{(\Delta r)^2}\right) + \frac{1}{r_i}\left(\frac{u_{i+1,j} - u_{i-1,j}}{2\Delta r}\right) + \frac{1}{r_i^2}\left(\frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{(\Delta\theta)^2}\right) + \Omega = -2,$$

for $i \in [2, n]$, $j \in [2, m]$, and where $\Omega = O((\Delta r)^2) + O((\Delta\theta)^2)$.

Further,

$$u_{i,j} = \frac{4r_i^2\Delta\theta^2\Delta r^2 + 2\Delta\theta^2 r_i^2\big(u_{i+1,j} + u_{i-1,j}\big) + \Delta r\Delta\theta^2 r_i\big(u_{i+1,j} - u_{i-1,j}\big) + 2\Delta r^2\big(u_{i,j+1} + u_{i,j-1}\big)}{4r_i^2\Delta\theta^2 + 4\Delta r^2}.$$

□

Question 4. *Generate difference equations corresponding to the boundary conditions. Ensure second order of convergence.*

Solution. 4 separate boundary conditions are required for the approximation of the PDE in polar coordinates, namely:

- **Boundary condition for the left vertical line.**

$$u_{1,j} = \sin(\theta_j), \quad \text{for} \quad j \in [1, m+1].$$

- **Boundary condition for the right vertical line.**

$$u_{n+1,j} = \cos(3\theta_j), \quad \text{for} \quad j \in [1, m+1].$$

- **Periodic boundary conditions.**
  - Condition, which guarantees that the horizontal line $\theta = 0$ coincides with the horizontal line $\theta = 2\pi$ in order to complete the two circles in the original annular region.

    $$u_{i,1} = u_{i,m+1}, \text{ for } i \in [1, n+1].$$

    The function used to calculate the values at the boundaries is as follows, for $i \in [2, n]$,

    $$u_{i,1} = \frac{4r_i^2\Delta\theta^2\Delta r^2 + 2\Delta\theta^2 r_i^2\big(u_{i+1,1} + u_{i-1,1}\big) + \Delta r\Delta\theta^2 r_i\big(u_{i+1,1} - u_{i-1,1}\big) + \dots}{4r_i^2\Delta\theta^2 + 4\Delta r^2}$$
    $$\dots + \frac{2\Delta r^2\big(u_{i,2} + u_{i,m}\big)}{4r_i^2\Delta\theta^2 + 4\Delta r^2} + \Omega = u_{i,m+1},$$

    where $\Omega = O((\Delta r)^2) + O((\Delta\theta)^2)$, i.e. second order of convergence. This is the formula for the interior nodes, with indices adjusted to approximate each node on the boundary with the 'fictitious nodes', which are actually interior nodes in the grid due to the periodic boundary conditions. Note that forward and backward finite difference formulas could be used as well, but they decrease the computation time, hence, lower the precision and accuracy of the numerical approximations as they disregard the step sizes in both the radial and azimuthal direction. In summary, this formula generates the boundary conditions as solutions of the given PDE, and is more precise as it takes into account the step sizes between the nodes, and hence produces smoother plots.
  - Condition, which guarantees that $u(r, \theta)$ is a C^1-smooth function, and its second-order of convergence difference equation is

    $$\big(u_{i,1}\big)_\theta = \frac{u_{i,2} - u_{i,m}}{2\Delta\theta} + O((\Delta\theta)^2) = \big(u_{i,m+1}\big)_\theta, \text{ for } i \in [1, n+1].$$

$\square$

Question 5. *Write the difference equations in the form of a Gauss-Jacobi iteration. Remember to list index ranges in the* MatLab *convention.*

Solution. In order to obtain the desired Gauss-Jacobi iterative scheme, the value of the function $u(r,\theta)$ at the i,j-th node should be expressed in terms of its neighbouring nodes, namely

$$\underbrace{\left(\frac{u_{i-1,j}-2u_{i,j}+u_{i+1,j}}{(\Delta r)^2}\right)+\frac{1}{r_i}\left(\frac{u_{i+1,j}-u_{i-1,j}}{2\Delta r}\right)+\frac{1}{r_i^2}\left(\frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{(\Delta\theta)^2}\right)+\Omega}_{\text{Common denominator: } 2r_i^2(\Delta r)^2(\Delta\theta)^2}=-2$$

$$\begin{aligned}
&2(\Delta\theta)^2r_i^2\big(u_{i-1,j}-2u_{i,j}+u_{i+1,j}\big)+\dots\\
&\qquad\dots+r_i(\Delta\theta)^2(\Delta r)\big(u_{i+1,j}-u_{i-1,j}\big)+\dots\\
&\qquad\dots+2(\Delta r)^2\big(u_{i,j+1}-2u_{i,j}+u_{i,j-1}\big)+\dots\\
&\qquad\dots+\underbrace{O((\Delta\theta)^4(\Delta r)^2)+O((\Delta r)^4(\Delta\theta)^2)}_{\text{Call this Ж}}=-4r_i^2(\Delta r)^2(\Delta\theta)^2.
\end{aligned}$$

Grouping the terms with respect to their node indices yields

$$\begin{aligned}
&u_{i,j}\Big(-4(\Delta\theta)^2r_i^2-4(\Delta r)^2\Big)+\dots\\
&\qquad\dots+u_{i-1,j}\Big(2(\Delta\theta)^2r_i^2-r_i(\Delta\theta)^2(\Delta r)\Big)+\dots\\
&\qquad\dots+u_{i+1,j}\Big(2(\Delta\theta)^2r_i^2+r_i(\Delta\theta)^2(\Delta r)\Big)+\dots\\
&\qquad\dots+u_{i,j}(\Delta r)^2+\text{Ж}=-4r_i^2(\Delta r)^2(\Delta\theta)^2.
\end{aligned}$$

Finally, the desired expression for $u_{i,j}$ is as follows:

$$\begin{aligned}
u_{i,j}=\frac{1}{4\Big(r_i^2(\Delta\theta)^2+(\Delta r)^2\Big)}\Bigg[&4(\Delta\theta)^2(\Delta r)^2r_i^2+\dots\\
&\dots+2(\Delta\theta)^2r_i^2\Big(u_{i+1,j}+u_{i-1,j}\Big)+\dots\\
&\dots+(\Delta\theta)^2(\Delta r)r_i\Big(u_{i+1,j}-u_{i-1,j}\Big)+\dots\\
&\dots+2(\Delta r)^2\Big(u_{i,j+1}+u_{i,j-1}\Big)+\text{Ж}\Bigg],
\end{aligned}$$

for $i\in[2,n]$, $j\in[2,m]$. □

Question 6. *function [r, t, u, niter] = PolarPoisson(n,m,tol).*

Solution. The listings below are the function and the main file respectively.

Listing 1: Gauss-Jacobi iteration with BCs for the Poisson equation.

```
%% 2019862s

%% This function obtains the numerical approximations
%% to the solutions of the given Poission equation
%% using the Gauss−Jacobi iterative method in polar
%% coordinates.

function [r,t,u,niter]=PolarPoisson(n,m,tol)

%% Parameters and Grid
% Step size in radial direction
deltaR=2/n;
% Step size in azimuthal direction
deltaTheta=(2*pi)/m;
% Radius goes from 1 to 3
r=1:deltaR:3;
% Theta goes from 0 to 2*pi
t=0:deltaTheta:(2*pi);
% Source term
f=-2;
% For loop, which populates the radial partition
ri=zeros(n-1,m-1);
for matDimR = 1:m-1
    ri(:,matDimR)=r(2:n)';
end;
%% Initial guess for Gauss−Jacobi
u=zeros(n+1,m+1);
uold=ones(n+1,m+1);
% Counter for the number of iterations
niter=0;

%% Gauss−Jacobi iteration based on FD equations
i=2:n;
j=2:m;
while norm(u-uold)>tol
    niter=niter+1;
    uold=u;
%%   Boundary conditions
    % Left vertical BC
    u(1,:)=sin(t);
    % Right vertical BC
    u(n+1,:)=cos(3*t);
    % Periodic BCs
    u(i,1)=((-2*f.*ri(:,1).^2.*deltaR.^2.*deltaTheta.^2)...
```

```
        +(2.*deltaTheta.^2.*ri(:,1).^2.*(u(i+1,1)+u(i-1,1)))...
        +(deltaR.*deltaTheta.^2.*ri(:,1).*(u(i+1,1)-u(i-1,1)))...
        +(2.*deltaR.^2.*(u(i,2)+u(i,m)))))./...
        (4.*ri(:,1).^2.*deltaTheta.^2+4.*deltaR.^2);
    u(i,m+1)=u(i,1);
%%   Internal Nodes
    u(i,j)=((-2*f.*ri.^2*deltaR.^2.*deltaTheta.^2)...
        +(2.*deltaTheta.^2.*ri.^2.*(u(i+1,j)+u(i-1,j)))...
        +(deltaR.*deltaTheta.^2.*ri.*(u(i+1,j)-u(i-1,j)))...
        +(2.*deltaR.^2.*(u(i,j+1)+u(i,j-1))))./...
        (4.*ri.^2.*deltaTheta.^2+4.*deltaR.^2);
end

%% Return solution
r
t
u=u(1:n+1,1:m+1)'
niter

end
```

Listing 2: Main file used to run the function.

```
%% 2019862s

%% Main file to run the PolarPoisson function,
%% produce a plot of the rectangular grid, convert to
%% polar coordinates, and plot the solution of the PDE.

[r,theta,u,niter]=PolarPoisson(30,33,10^(-5));

%% This figure plots the rectangular grid from r=1 to r=3, and
%% theta=0 to theta=2*pi in polar coordinates.
figure
surf(r,theta,u);
    title('Rectangular grid in polar coordinates')
    xlabel('Radius r')
ylabel('Angle theta')

%% This figure converts the rectangular grid in polar coordinates
%% back to Cartesian and plots the solutions, obtained by
%% the Gauss-Jacobi iteration for solving the PDE.
[r,theta]=meshgrid(r,theta);
[x,y]=pol2cart(theta,r);
figure
surf(x,y,u);
title('Numerical approximations of the solutions to the PDE')
xlabel('x')
ylabel('y')
```

Figure (4) below displays the output in the MATLAB console with $n = 4$ and $m = 5$, for simplicity.

```
>> mainFile

r =

    1.0000    1.5000    2.0000    2.5000    3.0000


t =

         0    1.2566    2.5133    3.7699    5.0265    6.2832


u =

         0    1.0970    1.5074    1.4446    1.0000
    0.9511    1.1876    0.9291    0.2481   -0.8090
    0.5878    1.2614    1.3594    1.0198    0.3090
   -0.5878    0.5819    0.9776    0.8509    0.3090
   -0.9511    0.0882    0.3113   -0.0252   -0.8090
   -0.0000    1.0970    1.5074    1.4446    1.0000


niter =

    37
```

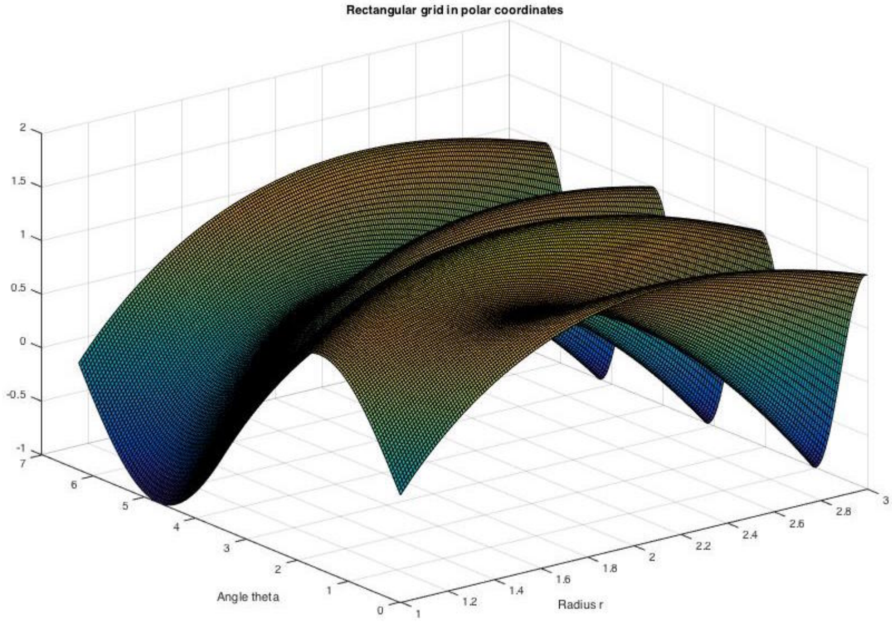
Figure 4: Console output demo.

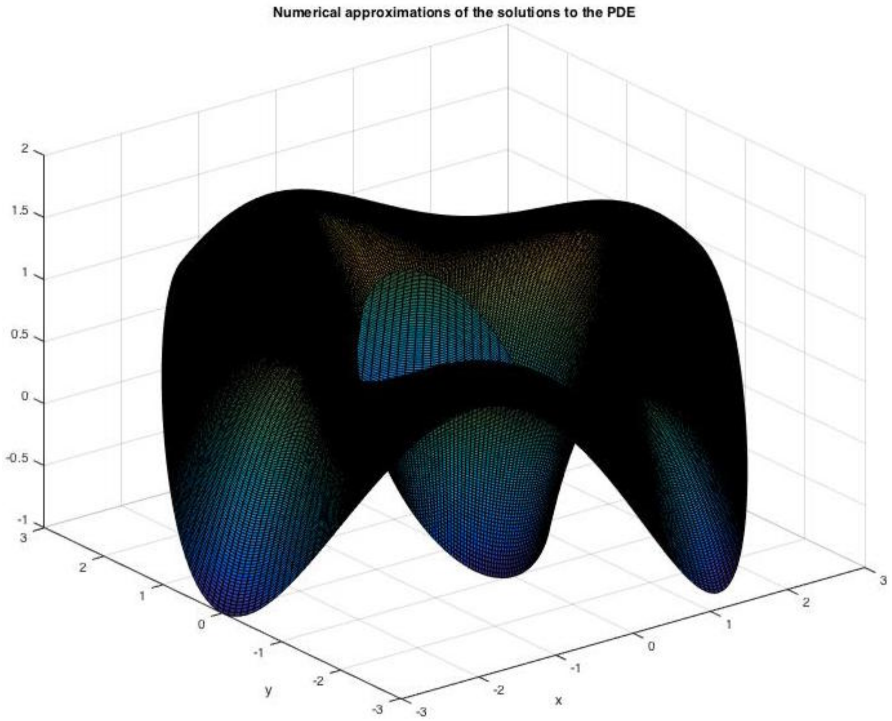
□

Question 7. *Run your* `PolarPoisson` *function to produce plots of the solution both in polar and Cartesian coordinates. The* MATLAB *plotting command* `surf` *may be used for plotting, while the command* `pol2cart` *may be used to convert from polar to Cartesian coordinates. Experiment with changing the boundary conditions and the non-homogeneous term in problem 1 to get other interesting solutions.*

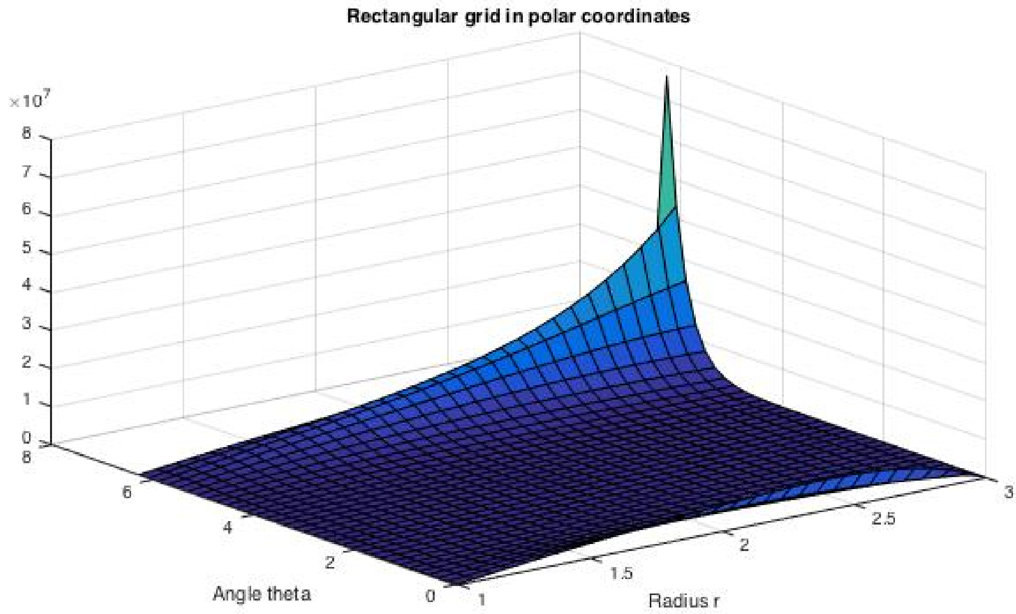
Solution. Several pairs of figures will be presented below. The first figure will be a plot of the solution in polar, and the second - a plot in Cartesian coordinates. A short description of the conditions used to plot will be provided before each pair is introduced. Note that the first pair took the longest time to be plotted as it required the largest number of iterations $niter = 182654$, but produced a much more smooth and precise plot.

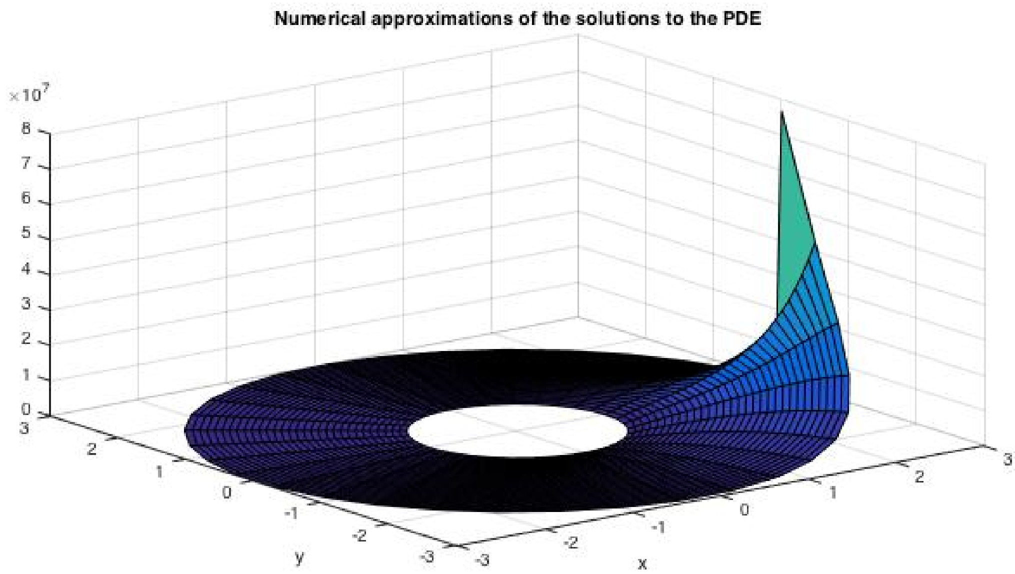
Graphs, produced by $n = 200$, $m = 200$, the specified tolerance is tol $= 10^{-15}$, $niter = 182654$. These two MATLAB figure files are included in the zip file.



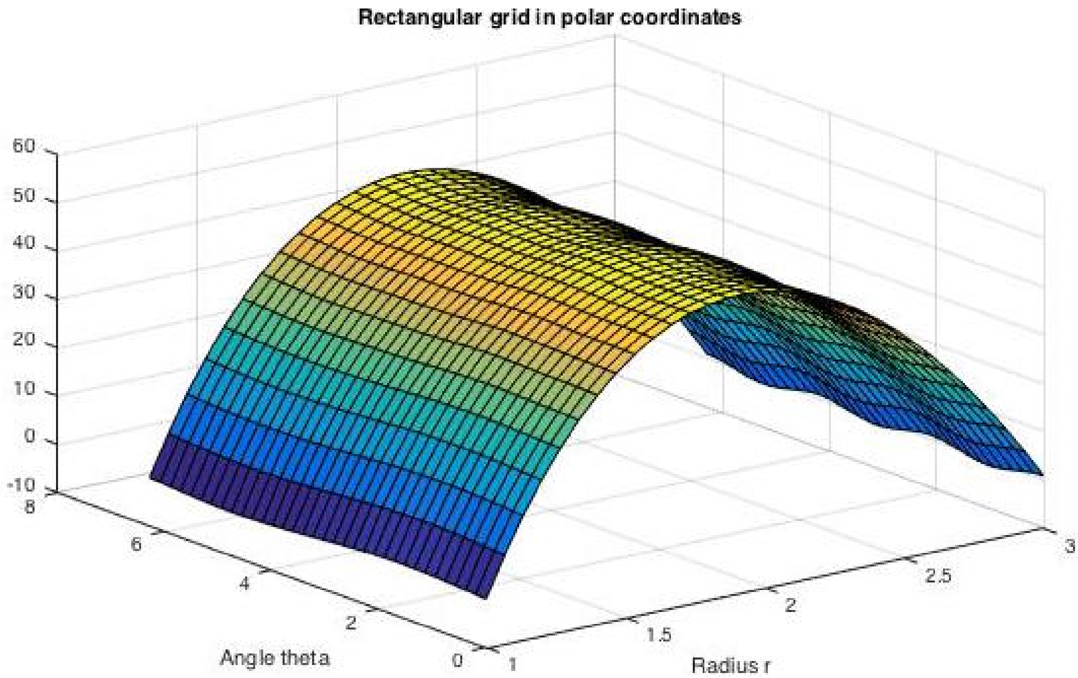


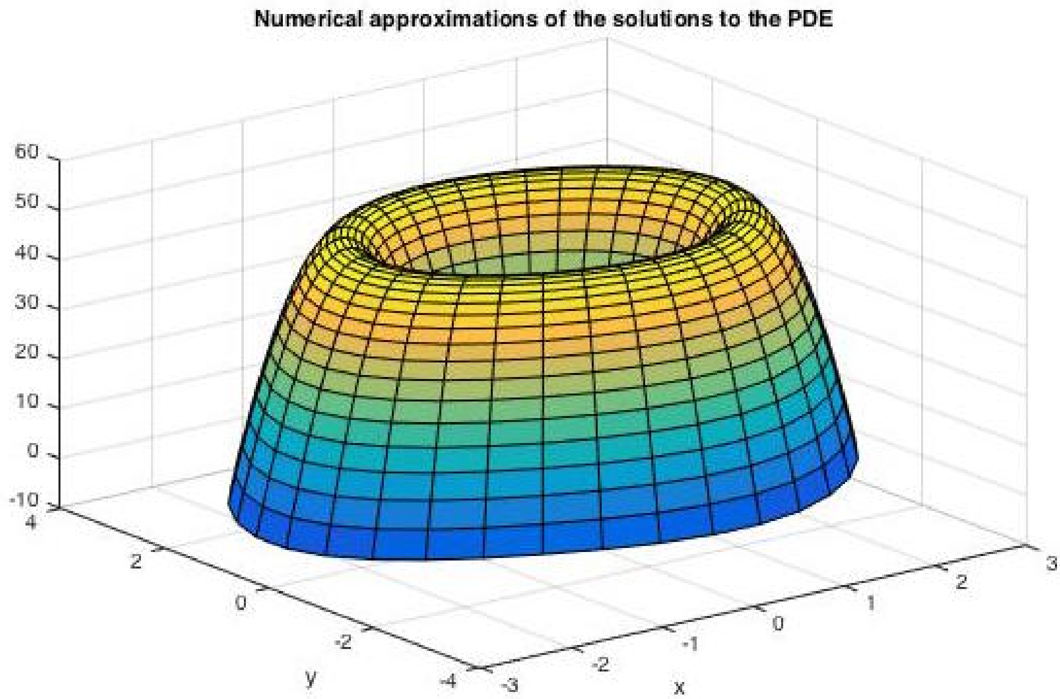
Graphs, produced by $n = 30$, $m = 33$, the specified tolerance is tol $= 10^{-5}$. The vertical conditions are changed to $u_{1,j} = \sinh(\theta)$ and $u_{n+1,j} = \cosh(3\theta)$, $niter = 4658$.



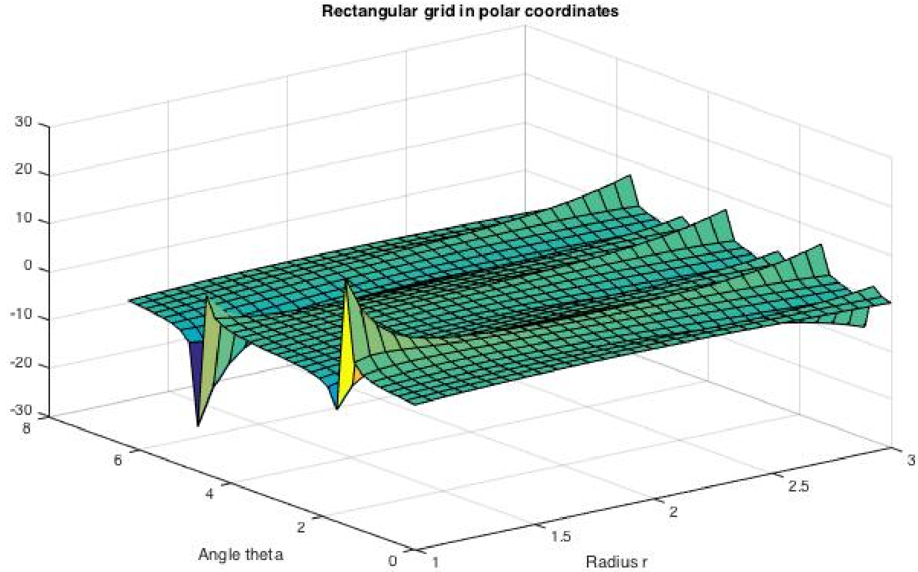


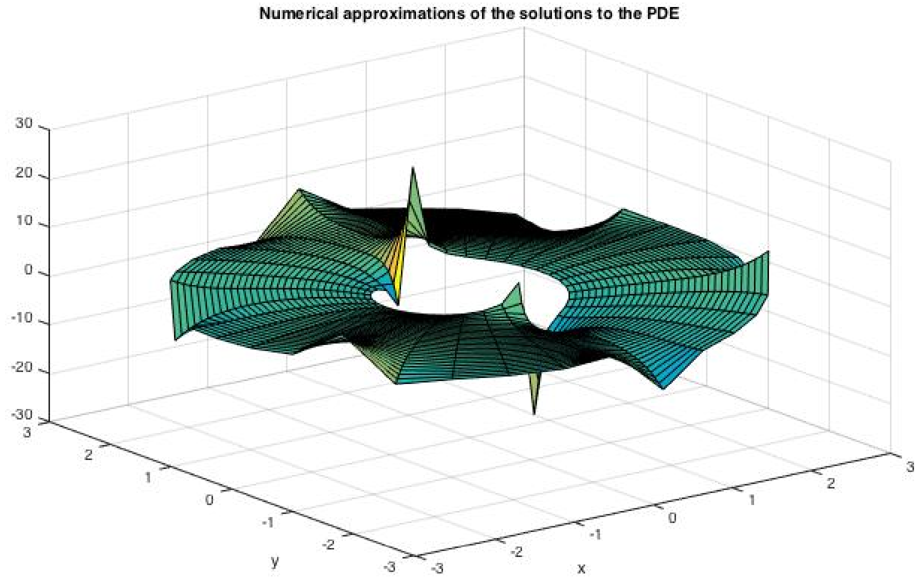
Graphs, produced by $n = 30$, $m = 33$, the specified tolerance is tol $= 10^{-5}$, and the non-homogeneous term was changed to $f = -100$, $niter = 2585$.





Graphs, produced by $n = 30$, $m = 33$, the specified tolerance is $\mathrm{tol} = 10^{-5}$, and the vertical conditions are changed to $u_{1,j} = \tan(\theta)$ and $u_{n+1,j} = \tan(3\theta)$, $niter = 1831$.





□