We cover some numerical methods for calculating approximations to the solution \( u(x, y, z, t) \) that satisfy the Diffusion Equation: 
\[
\sigma^2 \nabla^2 u = \frac{\partial u}{\partial t} - f
\]
a partial differential equation (PDE) with given boundary conditions on a specified region or body, where \( \sigma \) is a constant depending on the region or body, the Laplace operator is 
\[
\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2},
\]
and \( f \) is a function of \( u \), space, and time representing diffusion sources or sinks in the region or body of interest. This equation has many applications in engineering and Physics, for example fluid and heat flow, potential distribution, transmission lines, and others. This equation has been and is intensely studied, and there are many ways of attacking the calculation of solutions and approximating solutions to, most often, simplifications of the equation and of the region or body. We focus here on some of the key issues in calculating approximate numerical solutions. While very important, we do not cover the existence and uniqueness of solutions and we make the assumptions that appropriate continuity and differentiability conditions are met.

We first look at approximations based on breaking the region or body into discrete points and then making approximations to the derivatives for a method called Finite-Difference. The end result is a system of linear equations which when solved yields an approximation to the solution at the discrete points. We will cover the issues with this method, and how to obtain stable and accurate approximations. We address using irregular discrete points to improve efficiency and effectiveness of the process.

Finally we cover obtaining approximate solutions by calculating the best coefficients of a linear combination of functions. These are all variants of Galerkin methods. When the region or body of interest is partitioned into sub-sections or elements and we calculate the best coefficients in each element, we are using a Finite Element approach to approximate the solution.

**Derivative Approximations**

A key step in the Finite-Difference method is the approximation of partial derivatives. Let’s consider the calculation of the first and second partial derivatives: \( \frac{\partial u}{\partial x} \) and \( \frac{\partial^2 u}{\partial x^2} \) at a point based on values of the function \( u \) in the neighborhood of that point. In order to allow for irregular discrete points (variable stepsize), we will use polynomial fit through sufficient number of points. A polynomial form due to Lagrange is handy; from a set of \( n+1 \) points \( (x_0, u_0), (x_1, u_1), (x_2, u_2), \ldots, (x_n, u_n) \) the Lagrange polynomial of order \( n \) is given by:

\[
u(x) \approx \sum_{i=0}^{n} \left( \prod_{j=0}^{n} \frac{x - x_j}{x_i - x_j} \right) u_i
\]

For three points \( (x_{n-1}, u_{n-1}), (x_n, u_n), (x_{n+1}, u_{n+1}) \), the Lagrange polynomial is given by

\[
u(x) \approx \frac{(x - x_n)(x - x_{n+1})}{(x_{n-1} - x_n)(x_{n-1} - x_{n+1})} u_{n-1} + \frac{(x - x_{n-1})(x - x_{n+1})}{(x_n - x_{n-1})(x_n - x_{n+1})} u_n + \frac{(x - x_{n-1})(x - x_n)}{(x_{n+1} - x_{n-1})(x_{n+1} - x_n)} u_{n+1}
\]

Observe that this polynomial interpolates exactly at the given points: \( u(x_{n-1}) = u_{n-1}, u(x_n) = u_n, u(x_{n+1}) = u_{n+1} \)

We want to approximate the first and second partial derivatives. Differentiating the Lagrange polynomial, we obtain:

\[
\frac{\partial u}{\partial x} \approx \frac{(x - x_n) + (x - x_{n+1})}{(x_{n-1} - x_n)(x_{n-1} - x_{n+1})} u_{n-1} + \frac{(x - x_{n-1}) + (x - x_{n+1})}{(x_n - x_{n-1})(x_n - x_{n+1})} u_n + \frac{(x - x_{n-1}) + (x - x_n)}{(x_{n+1} - x_{n-1})(x_{n+1} - x_n)} u_{n+1}
\]

This first partial derivative formula evaluated at \( x = x_n \) becomes, after simplification and collecting terms,
If the stepsize is constant, \( h = x_{n+1} - x_n = x_n - x_{n-1} \), this formula becomes

\[
\frac{\partial u_n}{\partial x} \approx \frac{1}{2h} \left[ u_{n+1} - u_{n-1} \right]
\]

which is the central difference approximation to the derivative with error of order \( O(h^2) \). Using the partial derivative formula at \( x = x_{n-1} \) and \( x = x_{n+1} \) with constant stepsize yields the forward and backward difference approximations with error of order \( O(h^2) \) given by

\[
\frac{\partial u_{n-1}}{\partial x} \approx \frac{1}{2h} \left[ -3u_{n-1} + 4u_n - u_{n+1} \right] \quad \text{and} \quad \frac{\partial u_{n+1}}{\partial x} \approx \frac{1}{2h} \left[ u_{n-1} - 4u_n + 3u_{n+1} \right]
\]

The second partial derivative is constant over the interval of the three points because we derived a quadratic. The second partial derivative is as follows:

\[
\frac{\partial^2 u}{\partial x^2} \approx \frac{2}{x_{n+1} - x_{n-1}} \left[ \frac{1}{x_n - x_{n-1}} u_{n-1} - \frac{x_{n+1} - x_{n-1}}{(x_{n+1} - x_n)(x_n - x_{n-1})} u_n + \frac{1}{x_{n+1} - x_n} u_{n+1} \right]
\]

If the stepsize is constant, \( h = x_{n+1} - x_n = x_n - x_{n-1} \), this formula becomes

\[
\frac{\partial^2 u}{\partial x^2} \approx \frac{1}{h^2} \left[ u_{n+1} - 2u_n + u_{n-1} \right]
\]

the central difference second derivative approximation with error of order \( O(h^2) \)

While the equations become messier, obtaining partial derivative approximations using more than three points follows the same approach. We need to discuss briefly partial derivative approximations when two points are given, with one of the points also specifying a first partial derivative. That is, fitting a polynomial through the data \((x_{n-1}, u_{n-1}), (x_n, u_n, u'_n)\), where for convenience we are using prime notation for \( \frac{\partial u}{\partial x} \). This situation is important since boundary condition can specify first partial derivatives on the boundaries. We can fit a quadratic through the two points with the addition of the derivative, and this is called Hermite interpolation, and we will skip the details of the derivation and simply state the results:

\[
u(x) \approx \frac{(x - x_n)^2}{(x_n - x_{n-1})^2} u_{n-1} + \left[ 1 - \frac{(x - x_n)^2}{(x_n - x_{n-1})^2} \right] u_n + \frac{(x - x_n)^2}{x_n - x_{n-1}} u'_n
\]

\[
\frac{\partial u}{\partial x} \approx \frac{2(x - x_n)}{(x_n - x_{n-1})^2} u_{n-1} - \frac{2(x - x_n)}{(x_n - x_{n-1})^2} u_n + \left[ 1 + \frac{2(x - x_n)}{x_n - x_{n-1}} \right] u'_n
\]

\[
\frac{\partial^2 u}{\partial x^2} \approx \frac{2}{(x_n - x_{n-1})^2} \left[ u_{n-1} - u_n + (x_n - x_{n-1})u'_n \right]
\]

When the derivative specification is on the first point: \( (x_{n-1}, u_{n-1}, u'_{n-1}), (x_n, u_n) \), the equations are

\[
u(x) \approx \left[ 1 - \frac{(x - x_n)^2}{(x_n - x_{n-1})^2} \right] u_{n-1} + \frac{(x - x_{n-1})^2}{(x_n - x_{n-1})^2} u_n + \left[ x_{n-1} + \frac{(x - x_{n-1})^2}{x_n - x_{n-1}} \right] u'_{n-1}
\]

\[
\frac{\partial u}{\partial x} \approx -\frac{2(x - x_{n-1})}{(x_n - x_{n-1})^2} u_{n-1} + \frac{2(x - x_{n-1})}{(x_n - x_{n-1})^2} u_n + \left[ 1 - \frac{2(x - x_{n-1})}{x_n - x_{n-1}} \right] u'_{n-1}
\]
\[
\frac{\partial^2 u}{\partial x^2} \approx \frac{2}{(x_n - x_{n-1})^2} \left[ -u_{n-1} + u_n - (x_n - x_{n-1})u'_{n-1} \right]
\]

**Exercises for Derivative Approximations**

1) Show that the Lagrange polynomial
\[
u(x) \approx \frac{(x-x_n)(x-x_{n+1})}{(x_n-x_n)(x_n-x_{n+1})}u_{n-1} + \frac{(x-x_{n-1})(x-x_n)}{(x_{n-1}-x_n)(x_{n-1}-x_n)}u_n + \frac{(x-x_{n+1})(x-x_n)}{(x_{n+1}-x_n)(x_{n+1}-x_n)}u_{n+1}
\]
through the three points \((x_{n-1},u_{n-1}), (x_n,u_n), (x_{n+1},u_{n+1})\) interpolates precisely at the given points.

2) Show that using a constant stepsize \(h = x_{n+1} - x_n = x_n - x_{n-1}\) the first partial derivatives are given by
\[
\frac{\partial u_n}{\partial x} \approx \frac{1}{2h} \left[ u_{n+1} - u_{n-1} \right], \quad \frac{\partial u_{n-1}}{\partial x} \approx \frac{1}{2h} \left[ -3u_{n-1} + 4u_n - u_{n+1} \right], \quad \text{and} \quad \frac{\partial u_{n+1}}{\partial x} \approx \frac{1}{2h} \left[ u_{n-1} - 4u_n + 3u_{n+1} \right]
\]
and discuss when each one might be applied.

3) Discuss the suitability of a polynomial approximation through 4 points

4) Calculate the Lagrange polynomial through 5 points \((2,u_{n-2}), (3,u_{n-1}), (4,u_n), (5,u_{n+1}), (6,u_{n+2})\) approximating \(u(x)\). Find the first partial derivatives at \(x = 4\) \[
\left[ \frac{\partial u(4)}{\partial x} = \frac{1}{12} \left( u_{n-2} - 8u_{n-1} + 8u_{n+1} - u_{n+2} \right) \right], \quad \text{and the second partial derivative at } x = 4 \left[ \frac{\partial^2 u(4)}{\partial x^2} = \frac{1}{12} \left( -u_{n-2} + 16u_{n-1} - 30u_n + 16u_{n+1} - u_{n+2} \right) \right].
\]

**Finite Difference Method for Steady State**

Let’s consider the steady state problem for two dimensions and for a rectangular region to hopefully grasp the essence of the method and that its approach is not limited. Thus we will analyze the problem:
\[
\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y,u), \quad \text{with } 0 \leq x \leq 5, \quad 0 \leq y \leq 2, \quad u(x,y) = g(x,y) \quad \text{on the boundary}
\]

The administration of the data and presentation are the key elements of this method. If we used a fixed stepsize of \(\frac{1}{2}\) we are calculating 27 interior approximate discrete values of the solution. Accuracy is a function of the stepsizes, and for steady state the method is stable for any stepsize. Thus a good approach is to try fairly large stepsizes (coarse grids) at the start. Then reduce the stepsizes (finer grids) in those areas where the solution is changing the most. Let’s first try a coarse grid of 1 for each dimension as shown on the region chart on the right. This means we have 4 unknown discrete values of \(u\) that we are calculating:
\[
u(1,1) = u_{1,1}, \quad u(2,1) = u_{2,1}, \quad u(3,1) = u_{3,1}, \quad u(4,1) = u_{4,1}
\]

because we know the values on the boundary. These discrete values must satisfy the equation at each point, thus we have four equations that define the unknowns. We use the previously derived approximation to the second partial derivatives with constant stepsize \(h = 1\), and we obtain the four equations:
\[
(u_{0,1} - 2u_{1,1} + u_{2,1}) + (u_{1,0} - 2u_{1,1} + u_{1,2}) = f(1,1,u_{1,1})
\]
\[
(u_{1,1} - 2u_{2,1} + u_{3,1}) + (u_{2,0} - 2u_{2,1} + u_{2,2}) = f(2,1,u_{2,1})
\]
\[
(u_{2,1} - 2u_{3,1} + u_{4,1}) + (u_{3,0} - 2u_{3,1} + u_{3,2}) = f(3,1,u_{3,1})
\]
\[
(u_{3,1} - 2u_{4,1} + u_{5,1}) + (u_{4,0} - 2u_{4,1} + u_{4,2}) = f(4,1,u_{4,1})
\]

Using matrix notation and collecting terms, this system of equations is given by:
\[
\begin{bmatrix}
-4 & 1 & 0 & 0 \\
1 & -4 & 1 & 0 \\
0 & 1 & -4 & 1 \\
0 & 0 & 1 & -4
\end{bmatrix}
\begin{bmatrix}
u_{i,1} \\
u_{2,1} \\
u_{3,1} \\
u_{4,1}
\end{bmatrix}
= \begin{bmatrix}
f(1,1,1) - g(0,1) - g(1,0) - g(1,2) \\
f(2,1,1) - g(2,0) - g(2,2) \\
f(3,1,1) - g(3,0) - g(3,2) \\
f(4,1,1) - g(5,1) - g(4,0) - g(4,2)
\end{bmatrix}
\]

where we also used the given boundary condition function. If the function \(f(x, y, u)\) is linear on \(u\) then we can move the \(u\) term to the left side and solve the system of equations for the unknowns. If the function is not linear on \(u\) then we have to use an iterative method like Newton’s iteration to solve for the unknowns.

One may be able to deduce that the solution is changing the most on one end, and we will assume that for this problem it is on the left fifth of the region. We thus will use a stepsize of \(\frac{1}{5}\) there and use 1 for the rest, as the rectangle sketch shows. We now have 10 interior unknown points, and the equations are given by (Note that the partial derivative \(\frac{\partial^2 u}{\partial x^2}\) at \((1, 2/3)\) and at \((1,4/3)\) have irregular step sizes)

\[
\begin{align*}
4(u_{0,2/3} - 2u_{1/2,2/3} + u_{1,2/3}) + \frac{9}{4}(u_{1/2,0} - 2u_{1/2,1/3} + u_{1/2,4/3}) &= f(1/2, 2/3, u_{1/2,1/3}) \\
\frac{4}{3}(2u_{3/2,2/3} - 3u_{1,2/3} + u_{2,2/3}) + \frac{9}{4}(u_{1,0} - 2u_{1,2/3} + u_{1,4/3}) &= f(1, 2/3, u_{1,1/3}) \\
(u_{1,2/3} - 2u_{2,2/3} + u_{3,1/3}) + \frac{9}{4}(u_{2,0} - 2u_{2,2/3} + u_{2,4/3}) &= f(2, 2/3, u_{2,2/3}) \\
(u_{2,2/3} - 2u_{3,2/3} + u_{4,1/3}) + \frac{9}{4}(u_{3,0} - 2u_{3,2/3} + u_{3,4/3}) &= f(3, 2/3, u_{3,2/3}) \\
(u_{3,2/3} - 2u_{4,2/3} + u_{5,1/3}) + \frac{9}{4}(u_{4,0} - 2u_{4,2/3} + u_{4,4/3}) &= f(4, 2/3, u_{4,2/3}) \\
4(u_{0,4/3} - 2u_{1/2,4/3} + u_{1,4/3}) + \frac{9}{4}(u_{1/2,2} - 2u_{1/2,4/3} + u_{1,2,2}) &= f(1/2, 4/3, u_{1/2,4/3}) \\
\frac{4}{3}(2u_{3/2,4/3} - 3u_{1,4/3} + u_{2,4/3}) + \frac{9}{4}(u_{1,2} - 2u_{1,4/3} + u_{1,2}) &= f(1, 4/3, u_{1,4/3}) \\
(u_{1,4/3} - 2u_{2,4/3} + u_{3,4/3}) + \frac{9}{4}(u_{2,2} - 2u_{2,4/3} + u_{2,2}) &= f(2, 4/3, u_{2,4/3}) \\
(u_{2,4/3} - 2u_{3,4/3} + u_{4,4/3}) + \frac{9}{4}(u_{3,2} - 2u_{3,4/3} + u_{3,2}) &= f(3, 4/3, u_{3,4/3}) \\
(u_{3,4/3} - 2u_{4,4/3} + u_{5,4/3}) + \frac{9}{4}(u_{4,2} - 2u_{4,4/3} + u_{4,2}) &= f(4, 4/3, u_{4,4/3})
\end{align*}
\]

Using matrix notation and collecting terms, this system of equations is given by:

\[
\begin{bmatrix}
25/2 & 4 & 0 & 0 & 0 & 9/4 & 0 & 0 & 0 & 0 \\
8/3 & -17/2 & 4/3 & 0 & 0 & 0 & 9/4 & 0 & 0 & 0 \\
0 & 1 & -13/2 & 1 & 0 & 0 & 0 & 9/4 & 0 & 0 \\
0 & 0 & 1 & -13/2 & 1 & 0 & 0 & 0 & 9/4 & 0 \\
0 & 0 & 0 & 1 & -13/2 & 1 & 0 & 0 & 9/4 & 0 \\
9/4 & 0 & 0 & 0 & 0 & -25/2 & 4 & 0 & 0 & 0 \\
0 & 9/4 & 0 & 0 & 0 & 8/3 & -17/2 & 4/3 & 0 & 0 \\
0 & 0 & 9/4 & 0 & 0 & 0 & 1 & -13/2 & 1 & 0 \\
0 & 0 & 0 & 9/4 & 0 & 0 & 0 & 1 & -13/2 & 1 \\
0 & 0 & 0 & 0 & 9/4 & 0 & 0 & 1 & -13/2 & 1
\end{bmatrix}
\begin{bmatrix}
u_{1/2,1/3} \\
u_{1/2,2/3} \\
u_{1/2,3/3} \\
u_{1/2,4/3} \\
u_{1/2,5/3} \\
u_{1,2/3} \\
u_{1,3/3} \\
u_{1,4/3} \\
u_{1,5/3} \\
u_{2,2/3} \\
u_{2,3/3} \\
u_{2,4/3} \\
u_{2,5/3} \\
u_{3,2/3} \\
u_{3,3/3} \\
u_{3,4/3} \\
u_{3,5/3} \\
u_{4,2/3} \\
u_{4,3/3} \\
u_{4,4/3} \\
u_{4,5/3}
\end{bmatrix}
= \begin{bmatrix}
f(1/2,2/3,u_{1/2,1/3}) - (9/4)g(1/2,0) - 4g(0,2/3) \\
f(1/2,3,u_{1/2,2/3}) - (9/4)g(1,0) \\
f(2,2/3,u_{2,2/3}) - (9/4)g(2,0) \\
f(3,2/3,u_{3,2/3}) - (9/4)g(3,0) \\
f(4,2/3,u_{4,2/3}) - (9/4)g(4,0) - g(5,2/3) \\
f(1/2,4/3,u_{1/2,4/3}) - (9/4)g(1/2,2) - 4g(0,4/3) \\
f(1/2,3,u_{1/2,4/3}) - (9/4)g(1,2) \\
f(2,2/3,u_{2,4/3}) - (9/4)g(2,2) \\
f(3,2/3,u_{3,4/3}) - (9/4)g(3,2) \\
f(4,2/3,u_{4,4/3}) - (9/4)g(4,2) - g(5,4/3)
\end{bmatrix}
\]
where we also used the given boundary condition function. As an example, let \( f(x, y, u) = -(x + y + u) \), and \( g(5, y) = 5 \) and 0 elsewhere on the boundary, we obtain the values:

\[
\begin{bmatrix}
  u_{1/2,1/3} \\
  u_{1,2/3} \\
  u_{2,2/3} \\
  u_{3,2/3} \\
  u_{4,2/3} \\
  u_{1/2,4/3} \\
  u_{1,4/3} \\
  u_{2,4/3} \\
  u_{3,4/3} \\
  u_{4,4/3}
\end{bmatrix} =
\begin{bmatrix}
  .7059430086 \\
  1.296260508 \\
  2.263802513 \\
  3.137171380 \\
  4.09358765 \\
  7.851715123 \\
  1.401941822 \\
  2.378140118 \\
  3.250939841 \\
  4.110060072
\end{bmatrix}
\]

**Observations**

As the number of discrete points rises (finer grids), the administration of the system of equations becomes the key problem. Keeping the ordering of the unknown discrete points in an orderly manner helps deal with this problem. In addition, as the matrix size rises we want to keep the non-zero patterns as regular as possible, as this helps programming the problem.

Keeping the grid fine only over those regions where the solution is changing the most is key to keeping the system of equations from becoming very large. Grids can be like in the sketch at right, and for these types one needs to use forward and backward difference second derivative approximations given by:

\[
\frac{\partial^2 u}{\partial x^2} \approx \frac{1}{h^2} \left[ 2u_n - 5u_{n-1} + 4u_{n-2} - u_{n-3} \right], \quad \frac{\partial^2 u}{\partial x^2} \approx \frac{1}{h^2} \left[ 2u_n - 5u_{n+1} + 4u_{n+2} - u_{n+3} \right]
\]

with accuracy of order \( O(h^2) \). The following less accurate, order \( O(h) \), three point equations are less desirable:

\[
\frac{\partial^2 u}{\partial x^2} \approx \frac{1}{h^2} \left[ u_n - 2u_{n-1} + u_{n-2} \right], \quad \frac{\partial^2 u}{\partial x^2} \approx \frac{1}{h^2} \left[ u_n - 2u_{n-1} + u_{n-2} \right]
\]

Multigrid methods that automatically start from simple coarse grids and automatically move to finer grids improve efficiency and accuracy and are used in a number of computer programs.

**Exercises for Finite Difference Methods for Steady State**

1) Using 1/10 stepsize for a rectangle \( 0 \leq x \leq 10, \ 0 \leq y \leq 2 \), calculate the number of unknown points in the interior of such a region.

2) Use \( f(x, y, u) = -(x + y + u) \), and \( g(5, y) = 5 \) and 0 elsewhere on the boundary to calculate four unknown values of \( u \) for the first problem setup above that had fixed steps \( h = 1 \) in both the \( x \) and \( y \) directions.

3) Use the backward difference approximation \( \frac{\partial^2 u}{\partial x^2} \approx \frac{1}{h^2} \left[ 2u_n - 5u_{n-1} + 4u_{n-2} - u_{n-3} \right] \) to write the partial derivative at the point \( (2, \frac{1}{2}) \) for the grid sketched in the Observations section above.

**Finite Difference Method for Transient**

Applying the Finite Difference Method for transient problems requires approximating the first partial derivative \( \frac{\partial u}{\partial t} \) and we can take advantage of calculating the solution at a point in time using previous and current values (like an Initial Value Problem); in other words we don’t have to solve for all discrete time values at each space point all at one time. In order to see
the process and not become overwhelmed with keeping track of data, let’s consider the problem of approximating \( u(x, t) \) that solves:
\[
\frac{\partial^2 u}{\partial x^2} = \frac{\partial u}{\partial t}, \quad t \geq 0, \quad 0 \leq x \leq 1, \quad u(x, 0) = g(x), \quad u(0, t) = a(t), \quad u(1, t) = b(t)
\]
This problem could simulate the calculation of the temperature distribution of a thin rod (like in an electric oven) of length 1. The function \( u(0, t) = a(t) \) represents how the temperature is being held at \( x = 0 \), the function \( u(1, t) = b(t) \) represents how the temperature is being held at \( x = 1 \), and \( u(x, 0) = g(x) \) describes the initial temperature profile of the rod at the start of the simulation.

The partial derivative \( \frac{\partial^2 u}{\partial x^2} \) is approximated as we did before, namely with the central difference formula. However the discrete equations can be written from strictly the previous time step space data, from the current time step, or from a combination. Using the notation \( u(x_i, t_n) = u_{i,n} \) with fixed steps of \( h = x_i - x_{i-1} \) and \( k = t_n - t_{n-1} \), the approximate partial derivatives must satisfy our equation at each point:
\[
\frac{1}{h^2}[u_{i-1,n-1} - 2u_{i,n-1} + u_{i+1,n-1}] = \frac{1}{k}[u_{i,n} - u_{i,n-1}], \quad \text{each } u_{i,n} \text{ is defined from previous time values (explicit formulation)}
\]
\[
\frac{1}{h^2}[u_{i-1,n} - 2u_{i,n} + u_{i+1,n}] = \frac{1}{k}[u_{i,n} - u_{i,n-1}], \quad \text{each } u_{i,n} \text{ is defined using mostly current time values (implicit formulation)}
\]
and a combination of the above two formulations:
\[
\frac{\theta}{h^2}[u_{i-1,n} - 2u_{i,n} + u_{i+1,n}] + \frac{1 - \theta}{h^2}[u_{i-1,n-1} - 2u_{i,n-1} + u_{i+1,n-1}] = \frac{1}{k}[u_{i,n} - u_{i,n-1}]
\]
where the explicit formulation is for \( \theta = 0 \), the implicit formulation is for \( \theta = 1 \), and the Crank-Nicolson method is for \( \theta = 1/2 \).

The explicit formulation is the simplest to use as one can solve for every value \( u_{i,n} \) starting from the initial conditions at \( t_0 \) without solving a linear system of equations. The explicit formulation has accuracy of order \( O(h^2 + k) \) and implicit about the same. The problem with the explicit method is the requirement that \( k / h^2 < 1/2 \) for the problem we are considering; in general this means explicit methods may require very impractically small stepsizes. The safest methods to use are implicit methods because they are stable for any stepsize value. The Crank-Nicolson is preferred over the implicit formulation above since it is only slightly more complex and has better accuracy than explicit methods of order \( O(h^2 + k^2) \). Methods with \( \theta \neq 0 \) have the disadvantage of requiring the solution of a tri-diagonal linear system of equations:
\[
-\frac{\theta}{h^2}u_{i-1,n} + \left(\frac{1}{k^2} + \frac{2\theta}{h^2}\right)u_{i,n} - \frac{\theta}{h^2}u_{i+1,n} = \frac{1 - \theta}{h^2}u_{i-1,n-1} + \left(\frac{1}{k^2} - 2\frac{1 - \theta}{h^2}\right)u_{i,n-1} + \frac{1 - \theta}{h^2}u_{i+1,n-1}
\]
which for \( \theta = 1/2 \) and after multiplying by \( 2k \) becomes:
\[
-\frac{k}{h^2}u_{i-1,n} + 2\left(1 + \frac{k}{h^2}\right)u_{i,n} - \frac{k}{h^2}u_{i+1,n} = \frac{k}{h^2}u_{i-1,n-1} + 2\left(1 - \frac{k}{h^2}\right)u_{i,n-1} + \frac{k}{h^2}u_{i+1,n-1}
\]

**Observations**

Explicit numerical methods always have stability problems, and often it is very difficult to know ahead of time how small the stepsize needs to be. Implicit methods are attractive because they are stable for all stepsize values at the expense of solving a tri-diagonal system of linear equations at each time step. The tri-diagonal system of linear equations is diagonally dominant which means it can be solved without pivoting using straightforward Gaussian Elimination.

**Exercises for Finite Difference Methods for Transient**

1) Obtain the \( n \)th equation for \( \theta = 0 \) which is a purely explicit formulation (unknowns on the left side of the equation).
2) Obtain the \( n \)th equation for \( \theta = 1 \) which is a purely implicit formulation (unknowns on the left side of the equation).
3) Write a short paragraph on the background of the Crank-Nicolson method (an Internet search is ideal).
**Preliminaries for Galerkin Methods**

Using a linear combination of expansion functions to approximate the solution of a difficult problem can be viewed as obtaining an approximate analytical solution. Unlike numerical methods yielding approximate numerical values of the solution, a linear combination of expansion functions solution approximation has many of the advantages of an exact analytic solution and it is therefore an area that has much current research interest. These Galerkin methods are based on calculating the best approximation (a minimization problem) as measured by some expression representing the error.

Thus we seek to approximate the solution \( u \) by

\[
        u \approx \hat{u} = \sum_{k=1}^{n} c_k u_k
\]

where the linearly independent expansion functions \( u_k \) are given (selected judiciously), and the problem is to calculate the best coefficients \( c_k \) that minimize an error expression. The expansion functions influence the accuracy of the approximation and the ease of calculating the coefficients.

A useful technique is to transform the problem into two somewhat easier problems. For example given:

\[
    \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = f(x, y, z) \text{ in some region } \Omega, \text{ with conditions on the boundary } \partial \Omega
\]

We can instead solve the two problems:

\[
    \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2} = 0, \text{ with } v \text{ satisfying the conditions on the boundary } \partial \Omega, \text{ and}
\]

\[
    \frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2} = f(x, y, z), \text{ } w = 0, \text{ on the boundary } \partial \Omega.
\]

Then the solution of the original problem is \( u = v + w \). This transformation allows calculating approximate solutions focusing on appropriate strategies for each type of problem and covers a large variety of problems.

**Steady State Approximations using Galerkin Methods**

Let’s consider approximating the solution of a two dimensional problem given by

\[
    \nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0, \quad 0 \leq x \leq 1, \quad 0 \leq y \leq 2, \quad u = x^2 + y^2, \text{ on the boundary}
\]

Ideal candidates for expansion functions are harmonic functions since they satisfy Laplace’s equation \( \nabla^2 u = 0 \). We then find the best coefficients to match the boundary conditions. Expansion functions that are harmonic can be obtained from the real and imaginary parts of \((x + iy)^n\) and are well suited for this polynomial boundary condition.

\[
    (x + iy)^0 = 1
\]

\[
    (x + iy)^1 = x + iy
\]

\[
    (x + iy)^2 = (x^2 - y^2) + i(2xy)
\]

\[
    (x + iy)^3 = (x^3 - 3xy^2) + i(3x^2y - y^3)
\]

\[
    (x + iy)^4 = (x^4 - 6x^2y^2 + y^4) + i(4x^3y - 4xy^3)
\]

\[
    (x + iy)^5 = (x^5 - 10x^3y^2 + 5xy^4) + i(5x^4y - 10x^2y^3 + y^5)
\]

\[
    (x + iy)^6 = (x^6 - 15x^4y^2 + 15x^2y^4 - y^6) + i(6x^5y - 20x^3y^3 + 6xy^5),
\]

For example, if we suspect that even functions might be best, we select:

\[
    u_1 = 1, \quad u_2 = x^2 - y^2, \quad u_3 = x^4 - 6x^2y^2 + y^4, \quad u_4 = x^6 - 15x^4y^2 + 15x^2y^4 - y^6
\]

and our approximate solution is

\[
    \hat{u} \approx \sum_{k=1}^{n} c_k u_k = c_1 u_1 + c_2 u_2 + c_3 u_3 + c_4 u_4,
\]

where we now seek to calculate the best coefficient...
values. Since each expansion function is harmonic, \( \frac{\partial^2 \hat{u}}{\partial x^2} + \frac{\partial^2 \hat{u}}{\partial y^2} = 0 \) is satisfied, so all we need is to focus on the boundary \( 0 \leq x \leq 1, \ 0 \leq y \leq 2 \) where we require \( \hat{u} = x^2 + y^2 \). Thus ideally we want

\[
c_1 u_1 + c_2 u_2 + c_3 u_3 + c_4 u_4 = x^2 + y^2, \quad \text{on the boundary}
\]

Since we have four unknown coefficients, we can insist that this equation be satisfied at four distinct points on the boundary, say \((0, 2), (1, 0), (1, 1), (1, 2)\). We then have the four linear equations:

\[
\begin{bmatrix}
1 & -4 & 16 & -64 \\
1 & 1 & 1 & 1 \\
1 & 0 & -4 & 0 \\
1 & -3 & -7 & 117
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2 \\
c_3 \\
c_4
\end{bmatrix}
= \begin{bmatrix} 4 \\ 1 \\ 2 \\ 5 \end{bmatrix}
\]

which results in the coefficients: \( c = [1.83, -0.79, -0.0435, 0.00435] \). Calculating the largest error on the boundary, we obtain approximately 1.83 at the point (0,0). We actually lucked out that the linear system of equations ended up with a unique solution. Most of the time we seek to find the closest, or best approximation. One can view this problem by using more than 4 points and then minimize the error expression

\[
\text{Minimize } \varepsilon = \sum_{i=1}^{m} \left[ c_1 u_1(x_i, y_i) + c_2 u_2(x_i, y_i) + c_3 u_3(x_i, y_i) + c_4 u_4(x_i, y_i) - (x_i^2 + y_i^2) \right]^2
\]

where \( m \) is the number of points. This is a Least Squares approximation. We can also derive this Least Square approximation using inner product spaces with orthogonal projections (see [2]). But let’s continue and derive the equations resulting from solving the minimization problem. Recall that the minimum is when the gradient \( \nabla \varepsilon = \left[ \frac{\partial \varepsilon}{\partial c_1}, \frac{\partial \varepsilon}{\partial c_2}, \frac{\partial \varepsilon}{\partial c_3}, \frac{\partial \varepsilon}{\partial c_4} \right] \) is equal to zero. This condition yields the linear system of equations

\[
\begin{bmatrix}
\sum_{i=1}^{m} u_1^2 & \sum_{i=1}^{m} u_1 u_2 & \sum_{i=1}^{m} u_1 u_3 & \sum_{i=1}^{m} u_1 u_4 \\
\sum_{i=1}^{m} u_2 u_1 & \sum_{i=1}^{m} u_2^2 & \sum_{i=1}^{m} u_2 u_3 & \sum_{i=1}^{m} u_2 u_4 \\
\sum_{i=1}^{m} u_3 u_1 & \sum_{i=1}^{m} u_3 u_2 & \sum_{i=1}^{m} u_3^2 & \sum_{i=1}^{m} u_3 u_4 \\
\sum_{i=1}^{m} u_4 u_1 & \sum_{i=1}^{m} u_4 u_2 & \sum_{i=1}^{m} u_4 u_3 & \sum_{i=1}^{m} u_4^2
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2 \\
c_3 \\
c_4
\end{bmatrix}
= \begin{bmatrix} \sum_{i=1}^{m} u_1(x_i^2 + y_i^2) \\ \sum_{i=1}^{m} u_2(x_i^2 + y_i^2) \\ \sum_{i=1}^{m} u_3(x_i^2 + y_i^2) \\ \sum_{i=1}^{m} u_4(x_i^2 + y_i^2) \end{bmatrix}
\]

The limit as the number of points goes to infinity results in the line integral around the boundary \( \partial \Omega \) with the error expression becoming:

\[
\text{Minimize } \varepsilon = \int_{\partial \Omega} \left[ c_1 u_1 + c_2 u_2 + c_3 u_3 + c_4 u_4 - (x^2 + y^2) \right]^2 ds
\]

Since the boundary is a rectangle, the error expression becomes the sum of four line integrals. Recall that line integrals are done with the curves or “lines” parameterized. Therefore for this example the four line integrals are:

1) \( x = 0, \ y = t, \ 0 \leq t \leq 2, \ ds = \sqrt{\left( \frac{dx}{dt} \right)^2 + \left( \frac{dy}{dt} \right)^2} dt = dt \Rightarrow \int_0^2 \left[ c_1 u_1 + c_2 u_2 + c_3 u_3 + c_4 u_4 - t^2 \right]^2 dt
\]

2) \( x = t, \ y = 2, \ 0 \leq t \leq 1, \ ds = dt \Rightarrow \int_0^1 \left[ c_1 u_1 + c_2 u_2 + c_3 u_3 + c_4 u_4 - t^2 - 4 \right]^2 dt
\]
3) \( x = 1, \ y = t, \ 0 \leq t \leq 2, \ ds = dt \Rightarrow \int_0^2 \left[ c_1 u_1 + c_2 u_2 + c_3 u_3 + c_4 u_4 - 1 - t^2 \right]^2 dt \\

4) \( x = t, \ y = 0, \ 0 \leq t \leq 1, \ ds = dt \Rightarrow \int_0^1 \left[ c_1 u_1 + c_2 u_2 + c_3 u_3 + c_4 u_4 - t^2 \right]^2 dt \\

The minimum is when the gradient \( \nabla \epsilon = \left[ \frac{\partial \epsilon}{\partial c_1}, \frac{\partial \epsilon}{\partial c_2}, \frac{\partial \epsilon}{\partial c_3}, \frac{\partial \epsilon}{\partial c_4} \right] \) is equal to zero. This condition yields the linear system of equations (note that each line integral in the system of equations represents the sum of four integrals as shown above)

\[
\begin{bmatrix}
\int_{\Omega} u_1^2 ds \\
\int_{\Omega} u_2 u_1 ds \\
\int_{\Omega} u_3 u_1 ds \\
\int_{\Omega} u_4 u_1 ds \\
\int_{\Omega} u_2^2 ds \\
\int_{\Omega} u_2 u_2 ds \\
\int_{\Omega} u_3 u_2 ds \\
\int_{\Omega} u_4 u_2 ds \\
\int_{\Omega} u_3^2 ds \\
\int_{\Omega} u_2 u_3 ds \\
\int_{\Omega} u_3 u_3 ds \\
\int_{\Omega} u_4 u_3 ds \\
\int_{\Omega} u_4^2 ds
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2 \\
c_3 \\
c_4
\end{bmatrix}
= 
\begin{bmatrix}
\int_{\Omega} u_1(x^2 + y^2) ds \\
\int_{\Omega} u_2(x^2 + y^2) ds \\
\int_{\Omega} u_3(x^2 + y^2) ds \\
\int_{\Omega} u_4(x^2 + y^2) ds
\end{bmatrix}
\]

(note the similarity with the previous system of equations). Using Maple we obtain:

\[
\begin{bmatrix}
6 & -20 & 36 & 180 \\
-20 & 116 & -4244 & 180 \\
36 & -4244 & 7788 & -139796 \\
5 & 105 & -35 & 165 \\
180 & 180 & 139796 & 6699076 \\
7 & -7 & -165 & 1001
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2 \\
c_3 \\
c_4
\end{bmatrix}
= 
\begin{bmatrix}
12 \\
-132 \\
5 \\
188 \\
7 \\
1276 \\
9
\end{bmatrix}
\]

which yields \( c = [0.986, -0.963, -0.0655, 0.0054] \) and the largest error on the boundary is approximately \( 1.04 \) at the point \((1,0)\). Accuracy improves best when more expansion functions, or better expansion functions are used.

Galerkin’s method can also be applied to the second type of problem. For an example let’s consider the 2 dimensional problem

\[\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y) = xy, \ u = 0 \ \text{on} \ 0 \leq x, y \leq 1\]

For these problems we select expansion functions that satisfy the boundary condition of being zero, and then we calculate the best coefficients that satisfy the partial differential equation. We will use two expansion functions given by

\[
\hat{u} \approx \sum_{k=1}^{2} c_k u_k \quad \text{with} \quad u_1 = (x^2 - x)(y^2 - y), \quad u_2 = (x^3 - 3x^2/2 + x/2)(y^3 - 3y^2/2 + y/2)
\]

It can be shown that the boundary conditions are satisfied (exercise). We now need to define an appropriate error measure, and a suitable one is

\[
\text{Minimize } \epsilon = \iint (f - \nabla^2 u)^2 = \iint (f - c_1 \nabla^2 u_1 - c_2 \nabla^2 u_2)^2
\]

where the double integral is over the region where we want the equation satisfied. Notice that this is once again a Least Square approximation, which can be more generally derived with projections using inner product spaces. The minimum is when the gradient \( \nabla \epsilon = \left[ \frac{\partial \epsilon}{\partial c_1}, \frac{\partial \epsilon}{\partial c_2} \right] \) is equal to zero. This condition yields the linear system of equations
where $\nabla^2 u_i = 2\left[(y^2 - y) + (x^2 - x)\right]$, $\nabla^2 u_2 = \frac{3}{2}\left[(2x - 1)y(2y^2 - 3y + 1) + (2y - 1)x(2x^2 - 3x + 1)\right]$, and one example of the double integrals is $\iiint (\nabla^2 u_1)^2 = 4\int_0^1 \int_0^1\left[(y^2 - y) + (x^2 - x)\right]^2 \, dx \, dy = \frac{22}{45}$. The linear system of equations is given by

\[
\begin{bmatrix}
22/45 & 0 \\
0 & 17/1400
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2
\end{bmatrix}
= 
\begin{bmatrix}
-1/6 \\
-1/120
\end{bmatrix}
\]

which results in the coefficients: $c = [-15/44, -35/51]$, with a value for $\sqrt{\varepsilon} = \left(\iiint \left(f - c_1\nabla^2 u_1 - c_2\nabla^2 u_2\right)^2 \, dx \, dy\right)^{1/2} = 0.2204$.

Since the Laplace operator is linear one can use linear spaces using a function inner product to derive the best approximations. The two expansion functions chosen are orthogonal with the error chosen (a connection to the function inner product), which is why the system of linear equations we obtained was diagonal. With linear algebra we can define the Best Approximation to be orthogonal with the expansion functions instead of the above development which is effectively orthogonal to $\nabla^2 u$. Here is the linear algebra approach for Best Approximation.

Suitable inner products are

\[
\langle u_i, u_j \rangle = \int_0^1 \int_0^1 u_i \, u_j \, dx \, dy,
\]

and

\[
\langle u_i, -\nabla^2 u_j \rangle = \int_0^1 \int_0^1 u_i \left(-\nabla^2 u_j\right) \, dx \, dy.
\]

It can be shown using Green’s Theorem, that for zero valued boundary conditions (as we have here)

\[
\langle u_i, -\nabla^2 u_j \rangle = \left\langle -\nabla^2 u_i, u_j \right\rangle = \int_0^1 \int_0^1 \left(-\nabla^2 u_i\right) u_j \, dx \, dy = \int_0^1 \int_0^1 \left(\frac{\partial u_i}{\partial x} \frac{\partial u_j}{\partial x} + \frac{\partial u_i}{\partial y} \frac{\partial u_j}{\partial y}\right) \, dx \, dy.
\]

which means the operator $L = -\nabla^2$ with zero valued boundary conditions is self-adjoint with the defined inner product. We select expansion functions that satisfy the boundary condition of zero as before, and then we calculate the best coefficients that satisfy the partial differential equation. We will use two expansion functions given by

\[
\hat{u} \approx \sum_{k=1}^2 c_k u_k \quad \text{with} \quad u_1 = (x^2 - x)(y^2 - y), \quad u_2 = (x^3 - 3x^2/2 + x/2)(y^3 - 3y^2/2 + y/2).
\]

The Best Approximation is then $\left\langle -\nabla (u - \hat{u}), u_k \right\rangle = \langle u - \hat{u}, -\nabla u_k \rangle = 0, \ k = 1, 2$. which yields the linear system of equations

\[
\begin{bmatrix}
\left\langle -\nabla^2 u_1, u_1 \right\rangle & \left\langle -\nabla^2 u_1, u_2 \right\rangle \\
\left\langle -\nabla^2 u_2, u_1 \right\rangle & \left\langle -\nabla^2 u_2, u_2 \right\rangle
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2
\end{bmatrix}
= 
\begin{bmatrix}
\langle u_1, -xy \rangle \\
\langle u_2, -xy \rangle
\end{bmatrix}
\]

where $\nabla^2 u_1 = 2\left[(y^2 - y) + (x^2 - x)\right]$, $\nabla^2 u_2 = \frac{3}{2}\left[(2x - 1)y(2y^2 - 3y + 1) + (2y - 1)x(2x^2 - 3x + 1)\right]$, and one example of the inner product is $\left\langle -\nabla^2 u_1, u_1 \right\rangle = -\iiint u_1 \nabla^2 u_1 = -2\int_0^1 \int_0^1 (x^2 - x)(y^2 - y)[(y^2 - y) + (x^2 - x)] \, dx \, dy = \frac{1}{45}$. The linear system of equations is given by

\[
\begin{bmatrix}
1/45 & 0 \\
0 & 1/8400
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2
\end{bmatrix}
= 
\begin{bmatrix}
-1/144 \\
-1/14400
\end{bmatrix}
\]

which results in the coefficients: $c = [-5/16, -7/12]$, with a value for $\|\text{Error}\| = \left\langle \left(\int_0^1 \int_0^1 \left(\nabla^2 \hat{u} - xy\right)^2 \, dx \, dy\right)^{1/2} = 0.2216$.  

Approximating Solutions to Diffusion Equation, V2  
Al Jimenez, Cal Poly  
5/5/2011, 10 of 14 pages
The two expansion functions chosen are orthogonal with the inner product chosen, which is why the system of linear equations we obtained was diagonal. Substitution of these coefficients and expanding the polynomials, we obtain the approximate solution:

\[
\hat{u} = \frac{1}{24} \left( -14x^3y^3 + 21x^3y^2 - 7x^3y + 21x^2y^3 - 39x^2y^2 + 18x^2y - 7xy^3 + 18xy^2 - 11xy \right)
\]

The following graphs of the solution and absolute errors show the results. The largest absolute error is at \( x = y = 1 \).

**Transient Approximations using Galerkin Methods**

Less well known is that Galerkin’s method can be extended to some time dependent problems or initial value PDE. Let’s consider the two dimensional problem

\[
\frac{\partial u}{\partial t} = \nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}, \quad u(x, y, 0) = xy, \quad u(0, y, t) = u(x, 0, t) = u(x, 1, t) = u(1, y, t) = 0
\]

The problem is to find the solution \( u(x, y, t) \) in the specified region as a function of time that also satisfies the specified boundary conditions.

Galerkin’s method begins by the approximation of the solution with expansion functions of the following type

\[
u(x, y, t) \approx \hat{u}(x, y, t) = \sum_{j=1}^{n} v_j(t) u_j(x, y)
\]

Substitution into the PDE yields

\[
\frac{\partial u}{\partial t} = \sum_{j=1}^{n} v_j'(t) u_j(x, y) = \nabla^2 u = \sum_{j=1}^{n} v_j(t) \nabla^2 u_j(x, y)
\]

Focusing on satisfying the boundary conditions, one selects the \( u_j(x, y) \) such that they vanish when \( x = 0, \ x = 1, \ y = 0, \) or \( y = 1 \). This means the orthogonal expansion polynomials we just used are good candidates.

\[
u_1 = (x^2 - x)(y^2 - y), \quad u_2 = (x^3 - 3x^2/2 + x/2)(y^3 - 3y^2/2 + y/2)
\]

and thus our approximation to the solution is

\[
\hat{u}(x, y, t) = v_1 u_1 + v_2 u_2
\]

where the “coefficients” are now functions of time. We seek to satisfy the PDE, namely

\[
v_1' u_1 + v_2' u_2 \approx v_1 \nabla^2 u_1 + v_2 \nabla^2 u_2
\]
with \( \nabla^2 u_1 = 2[(y^2 - y) + (x^2 - x)] \), \( \nabla^2 u_2 = \frac{3}{2}[(2x - 1)y(2y^2 - 3y + 1) + (2y - 1)x(2x^2 - 3x + 1)] \). In practice we can only satisfying this equation approximately and seek the best values for \( v_1(t) \) and \( v_2(t) \) which are the coefficients as calculated in the last section.

We begin with the initial conditions at \( t = 0 \), and this means we desire the best approximation for

\[
\hat{u}(x, y, 0) = xy \approx v_1(0) u_1(x, y) + v_2(0) u_2(x, y), \quad \text{everywhere inside the region}
\]

Thus we calculate \( v_1(0) \) and \( v_2(0) \) which come the closest to satisfying this condition. We select the minimization of the error measure we used before, namely

\[
\text{Minimize } \epsilon = \int_0^1 \int_0^1 (v_1(0) u_1 + v_2(0) u_2 - xy)^2 \, dx \, dy
\]

This is the same problem we solved previously with \( v_1(0) \) and \( v_2(0) \) being the unknown coefficients. Again the minimum is when the gradient \( \nabla \epsilon = \left[ \frac{\partial \epsilon}{\partial v_1}, \frac{\partial \epsilon}{\partial v_2} \right] \) is equal to zero. This condition yields the linear system of equations

\[
\begin{bmatrix}
\iint (u_1)^2 \\
\iint (u_2)^2 \\
\iint u_1 u_2 \\
\iint u_2 u_1
\end{bmatrix}
\begin{bmatrix}
v_1(0) \\
v_2(0)
\end{bmatrix}
= \begin{bmatrix}
\iint u_1(xy) \\
\iint u_2(xy)
\end{bmatrix}
\]

and this linear system of equations becomes for this example

\[
\begin{bmatrix}
1/900 & 0 \\
0 & 1/705600
\end{bmatrix}
\begin{bmatrix}
v_1(0) \\
v_2(0)
\end{bmatrix}
= \begin{bmatrix}
1/144 \\
1/14400
\end{bmatrix}
\]

which results in the coefficients: \( v(0) = [25/4, 49] \) and the square root of the error is 0.2535. Therefore we begin with

\[
u(x, y, 0) = \frac{25}{4} u_1(x, y) + 49 u_2(x, y)
\]

This expands to:

\[
u(x, y, 0) \approx 49 x^3 y^3 - \frac{147}{2} x^3 y^2 + \frac{49}{2} x^3 y - \frac{147}{2} x^2 y^3 + \frac{233}{2} x^2 y^2 - 43x^2 y + \frac{49}{2} xy^3 - 43xy^2 + \frac{37}{2} xy
\]

The error norm for this 2-term Galerkin approximation is

\[
\| \text{Error} \| = \left\{ \int_0^1 \int_0^1 \left( u(x, y, 0) - f \right)^2 \, dx \, dy \right\}^{1/2} = 0.2536
\]

Here are 3D graphs of this result.
We now want to find the best approximation to the first order differential equation $v'_1u_1 + v'_2u_2 \approx v_1\nabla^2 u_1 + v_2\nabla^2 u_2$, seeking
the best coefficients $v_1(t)$ and $v_2(t)$ by solving

$$
\text{Minimize } E = \int_0^1 \int_0^1 (v'_1(t)u_1 + v'_2(t)u_2 - v_1\nabla^2 u_1 - v_2\nabla^2 u_2)^2 dxdy \text{, with initial conditions } v_1(0)=25/4 \text{ and } v_2(0)=49
$$

Again the minimum is when the gradient $\nabla E = \left[ \frac{\partial E}{\partial v_1}, \frac{\partial E}{\partial v_2} \right]$ is equal to zero. This time we obtain a difficult second order system of non-linear differential equations with this approach. Using best projections as covered in [2], we can take the inner product defined

$$
\langle g, h \rangle = \int_0^1 \int_0^1 g(x,y)h(x,y) dxdy
$$
on the desired equation $v'_1u_1 + v'_2u_2 \approx v_1\nabla^2 u_1 + v_2\nabla^2 u_2$ and use either the expansion functions $u_1$, and $u_2$ as shown in [2] or $-\nabla^2 u_1$, and $-\nabla^2 u_2$ which we use here (inner product symmetry is satisfied with this definition). The Best Approximation is obtained by finding the solution orthogonal to $-\nabla^2 u_1$ and $-\nabla^2 u_2$

$$
\langle v'_1u_1 + v'_2u_2 - v_1\nabla^2 u_1 - v_2\nabla^2 u_2, -\nabla^2 u_1 \rangle = 0 \Rightarrow \langle v'_1u_1, -\nabla^2 u_1 \rangle + \langle v'_2u_2, -\nabla^2 u_1 \rangle - \langle v_1\nabla^2 u_1, -\nabla^2 u_1 \rangle - \langle v_2\nabla^2 u_2, -\nabla^2 u_1 \rangle = 0
$$

$$
\langle v'_1u_1 + v'_2u_2 - v_1\nabla^2 u_1 - v_2\nabla^2 u_2, -\nabla^2 u_2 \rangle = 0 \Rightarrow \langle v'_1u_1, -\nabla^2 u_2 \rangle + \langle v'_2u_2, -\nabla^2 u_2 \rangle - \langle v_1\nabla^2 u_1, -\nabla^2 u_2 \rangle - \langle v_2\nabla^2 u_2, -\nabla^2 u_2 \rangle = 0
$$

In matrix form

$$
\begin{bmatrix}
\langle u_1, -\nabla^2 u_1 \rangle & \langle u_2, -\nabla^2 u_1 \rangle \\
\langle u_1, -\nabla^2 u_2 \rangle & \langle u_2, -\nabla^2 u_2 \rangle
\end{bmatrix}
\begin{bmatrix}
v'_1(t) \\
v'_2(t)
\end{bmatrix}
= 

\begin{bmatrix}
\langle \nabla^2 u_1, -\nabla^2 u_1 \rangle & \langle \nabla^2 u_1, -\nabla^2 u_2 \rangle \\
\langle \nabla^2 u_2, -\nabla^2 u_1 \rangle & \langle \nabla^2 u_2, -\nabla^2 u_2 \rangle
\end{bmatrix}
\begin{bmatrix}
v_1(t) \\
v_2(t)
\end{bmatrix}
$$

and using the inner product definition this matrix becomes the system of linear differential equations given by

$$
\begin{bmatrix}
\int u_1\nabla^2 u_1 & \int u_2\nabla^2 u_1 \\
\int u_1\nabla^2 u_2 & \int u_2\nabla^2 u_2
\end{bmatrix}
\begin{bmatrix}
v'_1(t) \\
v'_2(t)
\end{bmatrix}
= 

\begin{bmatrix}
\int (\nabla^2 u_1)^2 & \int \nabla^2 u_1\nabla^2 u_2 \\
\int \nabla^2 u_2\nabla^2 u_1 & \int (\nabla^2 u_2)^2
\end{bmatrix}
\begin{bmatrix}
v_1(t) \\
v_2(t)
\end{bmatrix}
$$

and for this example the matrix elements are given by

$$
\begin{bmatrix}
-1/45 & 0 \\
0 & -1/8400
\end{bmatrix}
\begin{bmatrix}
v'_1(t) \\
v'_2(t)
\end{bmatrix}
= 

\begin{bmatrix}
22/45 & 0 \\
0 & 17/1400
\end{bmatrix}
\begin{bmatrix}
v_1(t) \\
v_2(t)
\end{bmatrix}
$$

and because of orthogonality, we have uncoupled linear differential equations with solutions

$$
v_1(t) = \frac{25}{4} e^{-22t}, \text{ and } v_2(t) = 49 e^{-102t}
$$

which means our approximate solution to the PDE is

$$
u(x,y,t) \approx \frac{25}{4} e^{-22t}u_1 + 49e^{-102t}u_2, \text{ where }

u_1 = (x^2 - x)(y^2 - y) \\
u_2 = (x^3 - 3x^2 / 2 + x / 2)(y^3 - 3y^2 / 2 + y / 2)
$$

which expands to

$$
u(x,y,t) \approx \frac{25}{4} e^{-20t}[x(x-1)y(y-1)] + \frac{49}{4} e^{-84t}[(2x^3 - 3x^2 + x)(2y^3 - 3y^2 + y)]$$
Here are graphs at a few values of time:

Exercises for Galerkin Methods

1) The solution \( u(x, y, z) \) of the general problem 
\[
\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = f(x, y, z)
\]
inside some region with given boundary conditions can be derived from solutions of
\[
(1) \quad \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2} = 0 \quad v \text{ satisfies the boundary conditions, and}

(2) \quad \frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2} = f(x, y, z), \quad w = 0 \text{ on the boundary. Show that } u = v + w.

2) Show that the boundary conditions \( u = 0 \) on \( 0 \leq x, y \leq 1 \), when \( u = c_1 u_1 + c_2 u_2 \) for \( u_1 = (x^2 - x)(y^2 - y) \), and 
\[
u_2 = (x^3 - 3x^2/2 + x/2)(y^3 - 3y^2/2 + y/2)
\]
are satisfied.

3) Show that 
\[
\int_0^1 \int_0^1 (\nabla^2 u)(xy)dxdy = \frac{-1}{6}, \quad \text{for } \nabla^2 u_1 = 2\left[(y^2 - y) + (x^2 - x)\right]
\]

References

