Math 226C Project 1

Theodore Kwan
tmkwan@uci.edu

May 9, 2016

Contents

1 Project Description

This project analyzes the heat equation by solving it using the finite element method on a uniform grid, and on an adaptive spatial mesh.

Consider the parabolic partial differential equation on $\Omega = (0,1) \times (0,1)$:

$$\begin{cases} \frac{\partial u}{\partial t} - \Delta u = f(\mathbf{x}, t), & \mathbf{x} \in \Omega,\ t \in (0, T] \\ u(\mathbf{x}, 0) = u_0(\mathbf{x}), & \mathbf{x} \in \Omega \\ u(\mathbf{x}, t) = g_D(\mathbf{x}, t), & \mathbf{x} \in \partial\Omega,\ t \in [0, T] \end{cases} \tag{1}$$

This project analyzes the specific problem where the solution is given by:

$$u = \beta(t) \exp\left(-\left[(x - t + 0.5)^2 + (y - t + 0.5)^2\right] / 0.04\right) \tag{2}$$
$$= \beta(t) \exp\left(\alpha(x, y, t)\right) \tag{3}$$

Where:

$$\beta = 0.1\left[1 - \exp\left(-100(t - 0.5)^2\right)\right] = 0.1\left[1 - \exp\left(\gamma(t)\right)\right] \tag{4}$$

To find f, we have to calculate the left hand side of equation (1). This can be done by finding the partial derivatives of u:

$$\begin{aligned} \frac{\partial u}{\partial t} &= -0.1\gamma'(t) \exp\left[\alpha + \gamma\right] + \beta(t)\alpha_t(x, y, t) \exp(\alpha) \\ &= 20(t - 0.5) \exp\left[\alpha + \gamma\right] + 50\beta(t)\left[(x - t + 0.5) + (y - t + 0.5)\right] \exp(\alpha) \\ \alpha_t &= \frac{2}{0.04}\left[(x - t + 0.5) + (y - t + 0.5)\right],\ \gamma'(t) = -200(t - 0.5) \\ \frac{\partial u}{\partial x} &= -50\beta(t)\left[(x + t - 0.5)\right] \exp(\alpha) \end{aligned}$$

Which means:

$$\Delta u = -50\beta(t)\left[2 - \frac{2}{0.04}(x - t + 0.5)^2 - \frac{2}{0.04}(y - t + 0.5)^2\right] \exp(\alpha)$$

Thus:

$$\frac{\partial u}{\partial t} - \Delta u = 20(t - 0.5) \exp\left[\alpha + \gamma\right] \tag{5}$$
$$+ 50\beta(t)\left[1 + (x - t + 0.5) - 50(x - t + 0.5)^2\right] \exp(\alpha) \tag{6}$$
$$+ 50\beta(t)\left[1 + (y - t + 0.5) - 50(y - t + 0.5)^2\right] \exp(\alpha) \tag{7}$$

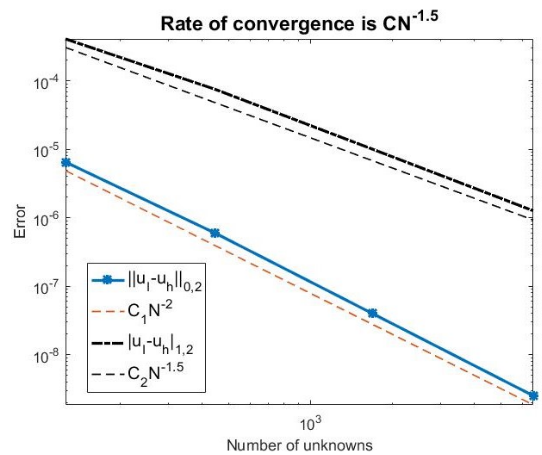
2 Part 1 - Uniform Mesh

On the uniform mesh, the Crank-Nicolson method was used along with mass lumping. All tests were done for $t \in [0, 2]$. The convergence rates were measured against:

$$N = \#\text{Nodes} + \frac{1}{\Delta t} \tag{8}$$

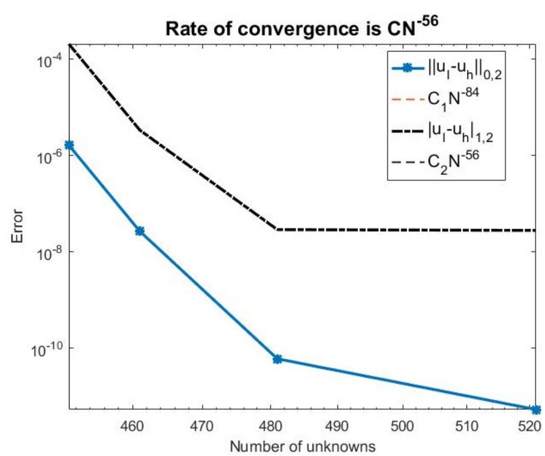
2.1 Convergence rate in space

When we vary h by uniform refinement and fix $\Delta t = 0.2$, we get the convergence rates:



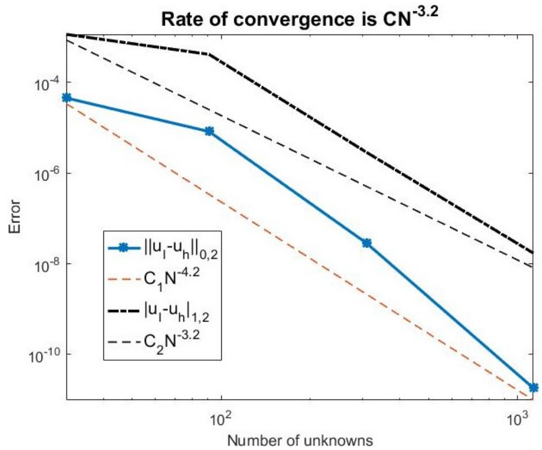
2.2 Convergence rate in time

When we vary Δt by dividing it by two and fix $h = 0.05$, we get the convergence rates:



2.3 Convergence rate in space and time

When we vary h through uniform refinement and vary Δt by division by two, we get the convergence rates:



3 Part 2 - Visualization

5 movies were recorded of the solution. One for the uniform grid, and the other four are for the different adaptive methods. They are located in the folder named Video. Each video is described in the list below.

1. HeatUniform.avi - Solving on a uniform grid using non-adaptive methods.
2. HeatAdaptiveRes1.avi - Solving on an adaptive mesh using the residual located in section (4.1.1)
3. HeatAdaptiveRes2.avi - Solving on an adaptive mesh using the residual located in section (4.1.2)
4. HeatAdaptiveRes3.avi - Solving on an adaptive mesh using the residual located in section (4.1.1) combined with the recovery residual.
5. HeatAdaptiveRecov.avi - Solving on an adaptive mesh using with the recovery residual in the ifem package.
6. HeatAdaptiveAlg2.avi - Solving on an adaptive mesh using algorithm 2 in section (4.2.2) with the residual in (4.1.1).

4 Part 4 - Adaptive Mesh

This section compares the adaptive finite element using the residuals from [2], [1], the estimaterecovery subroutine included in the ifem package and the combination of estimaterecovery and [2].

4.1 Residuals

The different residuals used are given in the following sections.

4.1.1 Residual 1

The residual from [2] is given by the following:

On interior edges e, define:

$$J_e^n = [\nabla u_\tau^n \cdot \mathbf{n}_e] \,, \quad \prod_\tau^n v = \frac{1}{(t_n - t_{n-1})|e|} \int_{t_{n-1}}^{t_n} \int_e v d\mathbf{x} dt \tag{9}$$

For any triangle $\tau \in \mathcal{T}$, define:

$$\prod_\tau^n v = \frac{1}{(t_n - t_{n-1})|\tau|} \int_{t_{n-1}}^{t_n} \int_\tau v d\mathbf{x} dt \tag{10}$$

Then the residuals are defined as:

$$(\eta_\tau^n)^2 = \int_{t_{n-1}}^{t_n} \left[|\tau| \left\| \textstyle\prod_\tau^n \left(f - \frac{\partial u_\tau}{\partial t}\right) \right\|_{0,\tau}^2 + \frac{1}{2} \sum_{e \in \mathcal{E}} |h_e| \left\| \textstyle\prod_e^n (J_e^n) \right\|_{0,e}^2 \right] dt \tag{11}$$

$$(\varepsilon_\tau^n)^2 = \int_{t_{n-1}}^{t_n} \left[|u_\tau^n - u_\tau^{n-1}|_{1,\tau}^2 \right] dt \tag{12}$$

For the purposes of the finite element method, all of the integrals in time were approximated using the trapezoid rule, and the integrals in space were approximated using the average value on an element and the method used in estimaterecovery by Professor Chen.

4.1.2 Residual 2

The residual from [1] is given by the description below.

On interior edges e, define:

$$J_e^n = [\nabla u_\tau^n \cdot \mathbf{n}_e] \tag{13}$$

Then for any triangle $\tau \in \mathcal{T}$, the residuals is defined as:

$$
\begin{aligned}
(\gamma_\tau^n)^2 &= |\tau|^2 \left\| \tfrac{u_\tau^n - u_\tau^{n-1}}{\Delta t} - \left(\theta f_\tau^{n+1} + (1-\theta) f_\tau^n\right) - \left(\theta \Delta u_\tau^n + (1-\theta)\Delta u_\tau^{n-1}\right) \right\|_{0,\tau}^2 \\
&\quad + \frac{1}{2} \sum_{e \in \mathcal{E}} |h_e| \|J_e^n\|_{0,e}^2 \\
(\varepsilon_\tau^n)^2 &= |\tau|^2 |u_\tau^n - u_\tau^{n-1}|_{1,\tau}^2 \\
(\eta_\tau^n)^2 &= (\gamma_\tau^n)^2 + (\varepsilon_\tau^n)^2
\end{aligned}
$$

4.1.3 Other Residuals

There were two other methods that were tried. These are:

1. Using the residual from estimaterecovery (known below as residual 4).
2. Combining the residuals for both estimate recovery and the residual contained in [2] (known below as residual 3).

4.2 Adaptive Finite Element Algorithms

For the adaptive finite element, two different algorithms were used. The first algorithm did both refinement and coarsening, and the second algorithm was a refine only algorithm.

4.2.1 Algorithm 1

Start with a step size Δt, the triangulation $\mathcal{T}_{n-1}$ obtained from the previous step and the solution u_{n-1} from the previous step.

Set $\mathcal{T}_n^1 := \mathcal{T}_{n-1}$, then while $i <$ max iterations

1. Solve the discrete problem for u_n on $\mathcal{T}_n^i$.
2. Using the data from u_n and u_{n-1}, compute η_n.
3. Mark elements for refinement or coarsening
4. The refinement or coarsening step has two possible outcomes:
   - If the number of nodes is less than the maximum number of nodes allowed, then refine the mesh to get $\mathcal{T}_n^{i+1}$.
   - If the number of nodes is greater than the maximum number of nodes allowed, coarsen the mesh to get $\mathcal{T}_n^{i+1}$.

4.2.2 Algorithm 2

Start with a step size Δt, the triangulation $\mathcal{T}_{n-1}$ obtained from the previous step and the solution u_{n-1} from the previous step.

Set $\mathcal{T}_n^1 := \mathcal{T}_{n-1}$, then while $i <$ max iterations and the number of nodes is less than the maximum number of nodes

1. Solve the discrete problem for u_n on $\mathcal{T}_n^i$.
2. Using the data from u_n and u_{n-1}, compute η_n.
3. Mark elements for refinement.
4. If the number of nodes is less than the maximum number of nodes allowed, then refine the mesh to get $\mathcal{T}_n^{i+1}$.

If the maximum number of nodes has already been reached, then solve the discrete problem for u_n on $\mathcal{T}_{n-1}$ and return.

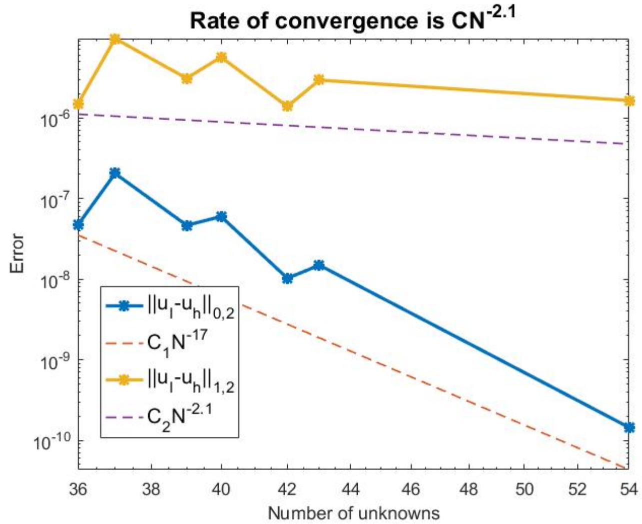
4.3 Convergence Results

For the different methods, each has a different convergence rate which is close to $|u_h - u|_{1,\Omega} \leq N^{1/2}$ for the first few steps. Then, the changes on the second step. The comparisons are located in the following pages.

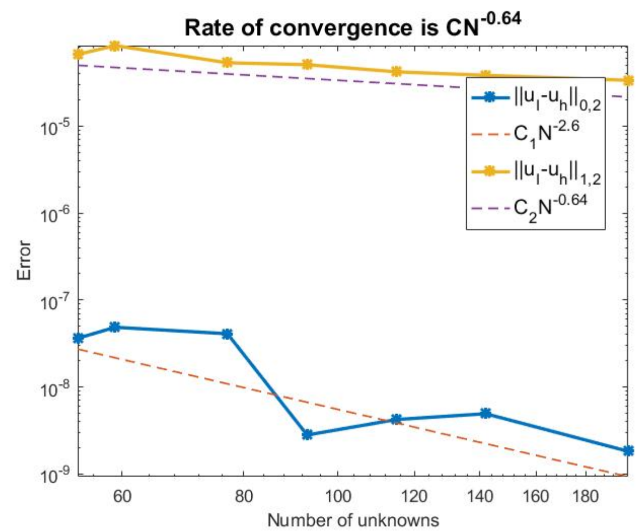
4.3.1 Algorithm 1 and Residual 1

The first residual converges initially at a much higher rate than $N^{-1/2}$ in the H_1 norm, and above N^{-15} in the L_2 norm. On the second step, the convergence rate for the error in the H_1 norm decreases while staying above $N^{-1/2}$, and the error in the L_2 norm converges much slower at a rate of N^{-2}. This is shown in the images below.

For $t = 0.1$:



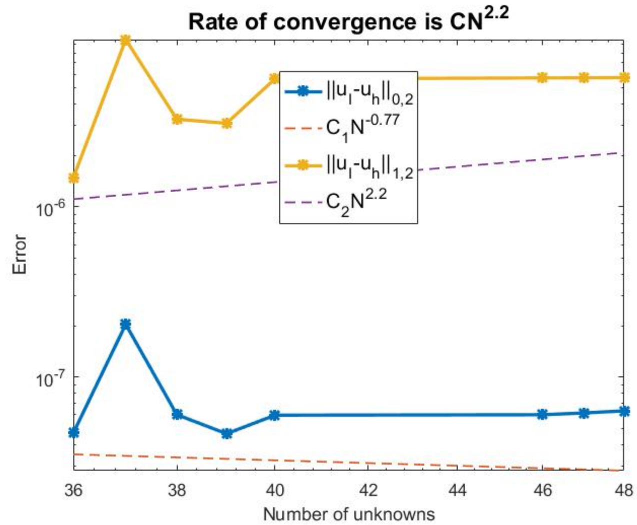
At $t = 0.2$:



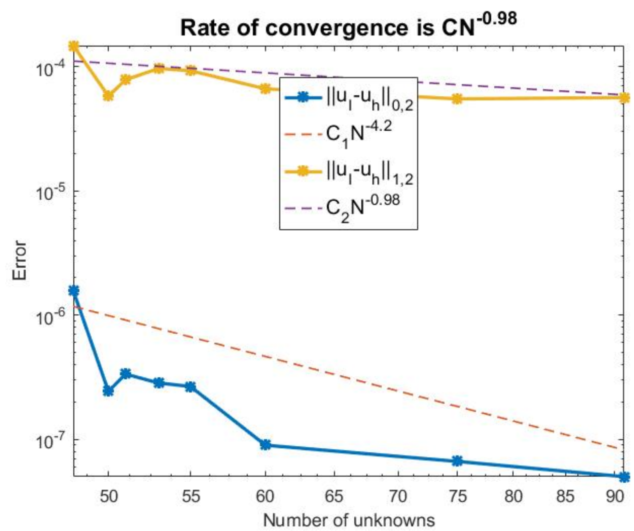
4.3.2 Algorithm 1 and Residual 2

The error when using the second residual increases initially at a rate of $N^{2.2}$ in the H_1 norm, and decreases slightly in the L_2 norm. On the second step, the H_1 error decreases at a higher rate and the L_2 norm does as well. This is shown in the images below:

At $t = 0.1$:



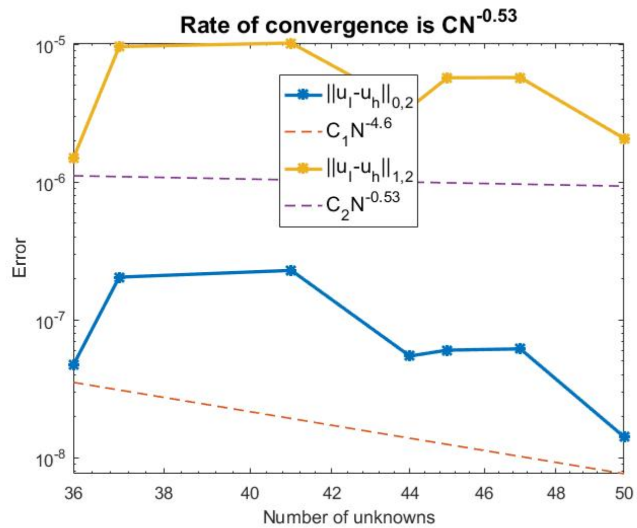
At $t = 0.2$:



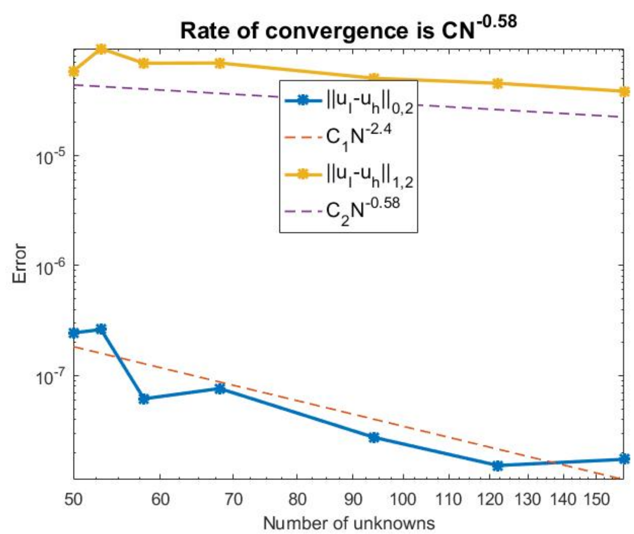
4.3.3 Algorithm 1 and Residual 3

The error obtained when using the third residual converges initially slightly higher than the expected rate of $N^{-1/2}$ in the H_1 norm, and a rate higher than N^{-2} in the L_2 norm. On the second step, the H_1 error decreases at approximately the same rate, while the L_2 norm decreases at a slower rate. This is shown in the images below:

At $t = 0.1$:



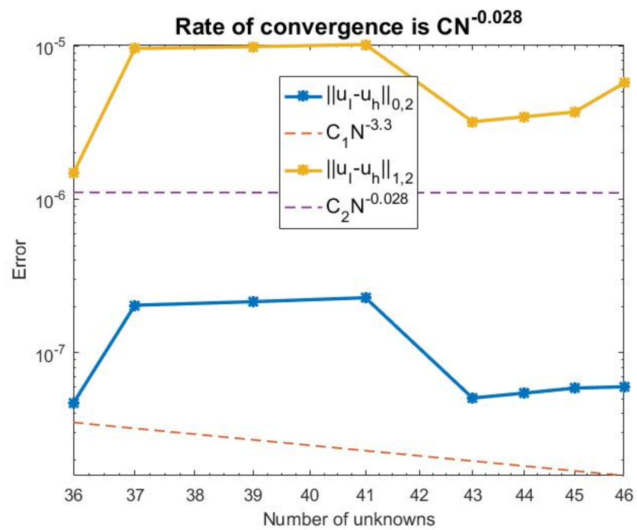
At $t = 0.2$:



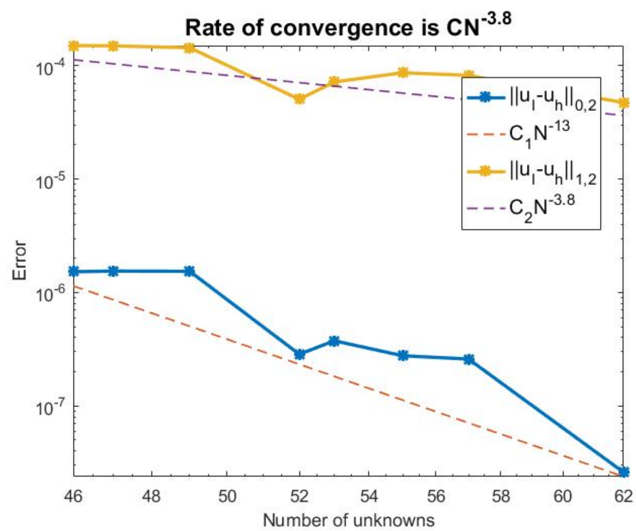
4.3.4 Algorithm 1 and Recovery Residual

The error using the recovery residual converges initially at a rate lower than $N^{-1/2}$ in the H_1 norm, and a high rate in the L_2 norm, then decreases at the much higher rates N^{-4} in the H_1 norm and N^{-13} in the L_2 norm. This is shown in the images below.

At $t = 0.1$:



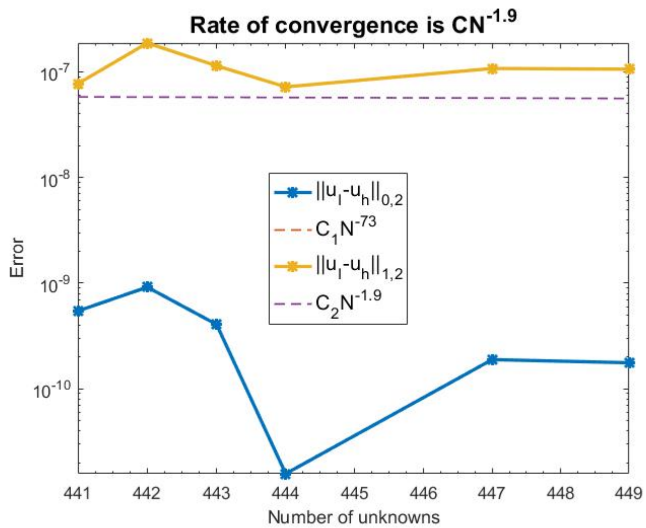
At $t = 0.2$:



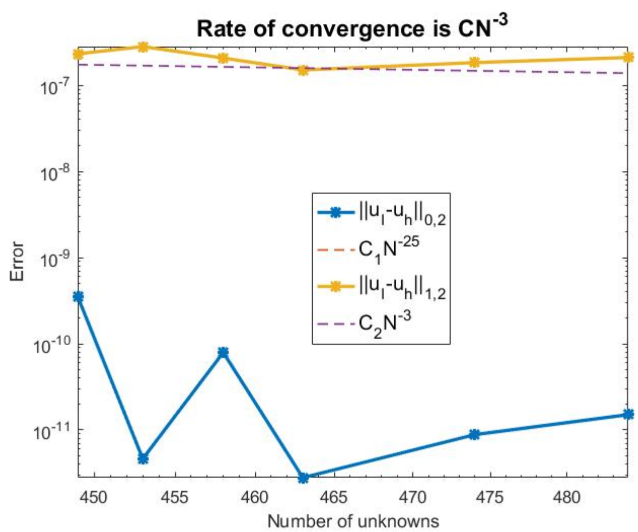
4.3.5 Algorithm 2 and Residual 1

The second algorithm using the first residual converges quickly in the H_1 norm, and at an extremely high rate of N^{-73} in the L_2 norm. On the second step, the error in the H_1 norm decreases at a higher rate than the first step, but the error in the L_2 norm decreases at about $\frac{1}{3}$ of the speed. This is shown in the images below.

At $t = 0.025$:



At $t = 0.05$:



4.4 Final Error

This section compares the error at the final timestep for the case where the singulary is present, and where the singularity is no longer present. In addition, it compares this to the error when the starting time is higher so that the singularity is present at the beginning of the refinement.

4.4.1 Long Final Time

When we solve the equation on the interval $t \in [0, 1.8]$ we get the following results for the error at the final step:

L_2 Error	H_1 Error	θ	N	Algorithm	Residual	Δt
4.25×10^{-8}	0.0059	0.6	5445	1	1	0.1
6.87×10^{-8}	0.0075	0.4	5381	1	2	0.1
5.32×10^{-8}	0.0047	0.6	5896	1	3	0.1
3.26×10^{-7}	0.0043	0.6	5462	1	4	0.1
1.51×10^{-8}	2.78×10^{-5}	NA	10201	Unif	0	0.1
3.75×10^{-13}	5.46×10^{-5}	0.4	7850	2	1	0.025
2.21×10^{-12}	2.61×10^{-5}	NA	10201	Unif	0	0.025

For the first algorithm, we can see that the different residuals give slightly different results, but the best result comes from the combined residual and recovery error estimator, as it minimizes the H_1 norm and L_2 norms collectively the most. The advantages of all of these algorithms is that they are obtaining similar L_2 error while only using around half of the number of nodes.

In comparison, the second algorithm decreases the error in the L_2 norm below that of the uniform grid, and decreases the error in the H_1 norm to be only slightly higher than the uniform grid. This method requires 2000 less nodes in order to work.

4.4.2 Short Final Time

When we solve the equation on the interval $t \in [0, 1.4]$ we get the following results for the error at the final step:

L_2 Error	H_1 Error	θ	N	Algorithm	Residual	Δt
1.8405×10^{-7}	0.0167	0.6	5539	1	1	0.1
1.7623×10^{-8}	0.0151	NA	10201	Uniform	0	0.1
1.5544×10^{-12}	0.0142	0.4	7850	2	1	0.025
2.409×10^{-9}	0.0151	NA	10201	Uniform	0	0.025

From the second table, we can see that the adaptive methods have a major advantage when the singularity is still active. Both have a similar H_1 and L_2

error while using less nodes. The second algorithm still performs very well (especially in regards to the L_2 norm). Intuitively, one would expect that the algorithm which moves the mesh to match the singularity would perform better, but this was not observed.

4.4.3 Different Starting Time

When we solve the equation on the interval $t \in [0.6, 1.8]$ we get the following results for the error at the final step:

L_2 Error	H_1 Error	θ	N	Algorithm	Residual	Δt
3.56×10^{-6}	0.0801	0.6	5409	1	1	0.1
1.23×10^{-6}	0.0831	0.4	5278	1	2	0.1
1.89×10^{-7}	0.0803	0.6	6409	1	3	0.1
1.95×10^{-7}	0.0793	0.6	6367	1	4	0.1
1.82×10^{-6}	0.0778	0.4	7483	2	1	0.0.025
2.47×10^{-6}	0.0775	NA	10201	Uniform	0	0.1
2.40×10^{-6}	0.0775	NA	10201	Uniform	0	0.025

The recovery error estimator and the combined error estimator using algorithm 1 give the best final error in this case. They have an H_1 error which is close to the uniform grid, as well as an L_2 error which is one order of magnitude lower than the uniform grid. This is done with less nodes in both cases. The second algorithm gives a slightly lower error in the H_1 norm, and an error in the L_2 norm which does not justify the extra nodes required. These results show the advantages of the moving adaptive algorithm to the uniform grid and the adaptive algorithm which only refines until it reaches the maximum number of nodes.

References

[1] "Numerical Treatment of Partial Differential Equations: Translated and revised by Martin Stynes". In: Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. Chap. Finite Element Methods for Unsteady Problems, pp. 317–373. ISBN: 978-3-540-71584-9. DOI: 10.1007/978-3-540-71584-9_5. URL: http://dx.doi.org/10.1007/978-3-540-71584-9_5.

[2] Marco Picasso. "Adaptive finite elements for a linear parabolic problem". In: *Computer Methods in Applied Mechanics and Engineering* 167.34 (1998), pp. 223–237. ISSN: 0045-7825. DOI: http://dx.doi.org/10.1016/S0045-7825(98)00121-2. URL: http://www.sciencedirect.com/science/article/pii/S0045782598001212.