Project Report

For

Nonlinear Diffusion Equation

Course INF 5620/9620 -Numerical Methods for the Partial Differential Equation

Hao, Zhao

Department of Geoscience & Department of Informatics, University of Oslo

Nov 28, 2015

1. Introduction

This project is to discuss various numerical aspects of a nonlinear diffusion model:

$$\rho u_t = \nabla.(\alpha(u)\nabla u) + f(x,t) \tag{1}$$

With initial condition u(x, 0) = I(x) and boundary condition ∂u/∂n = 0. The coefficient ρ is constant and α (u) is a known function of u.

2. Mathematical Problem

a) Introduce some finite difference approximation in time that leads to an implicit scheme (say a Backward Euler, Crank-Nicolson, or 2-step backward scheme). Derive a variational formulation of the initial condition and the spatial problem to be solved at each time step (and at the first time step in case of a 2-step backward scheme).

Based on the backward Euler scheme, we can write the nonlinear diffusion equation (1) as:

$$\rho(\frac{u^n - u^{n-1}}{\Delta t}) = \nabla.(\alpha(u^n)\nabla u^n) + f(x,t) \tag{2}$$

It yields:

$$u^{n-1} = u^n - \frac{\Delta t}{\rho}.\nabla.(\alpha(u^n)\nabla u^n) - \frac{\Delta t}{\rho} f(x,t) \tag{3}$$

We can rewrite the equation (3) as following, based on the notation rules used in the course:

$$u^{-1} = u - \frac{\Delta t}{\rho}.\nabla.(\alpha(u)\nabla u) - \frac{\Delta t}{\rho} f(x,t) \tag{4}$$

Based on the principle of Galerkin method: (R,v) = 0, ∀ v∈ V:

$$\int_\Omega (uv - \frac{\Delta t}{\rho}.\nabla.(\alpha(u)\nabla u)v - \frac{\Delta t}{\rho} f(x,t)\, v - u^{-1}v)dx = 0 \tag{5}$$

According to integration by part,:

$$\int_\Omega \frac{\Delta t}{\rho}.\nabla(\alpha(u)\nabla u).vdx = \frac{\Delta t}{\rho}.\big(\alpha(u(1))\nabla u(1)\big).v(1) - \frac{\Delta t}{\rho}.\big(\alpha(u(0))\nabla u(0)\big).v(0) - \int_\Omega \frac{\Delta t}{\rho}.\alpha(u)\nabla u.\nabla v dx \tag{6}$$

Considering the homogeneous Neumann conditions (∂u/∂n = 0)

$$\int_\Omega \frac{\Delta t}{\rho}.\nabla(\alpha(u)\nabla u).vdx = -\int_\Omega \frac{\Delta t}{\rho}.\alpha(u)\nabla u.\nabla v dx \quad (7)$$

Combine equation (5) and (7), we derived the variational form:

$$\int_\Omega (uv + \frac{\Delta t}{\rho}.\alpha(u)\nabla u.\nabla v)dx = \int_\Omega (u^{-1}v + \frac{\Delta t}{\rho}f(x,t)v)dx \quad (8)$$

Thus equation (8) is the variational form of nonlinear diffusion equation. By solving this equation on each time level, we will derive the solution of this nonlinear PDE.

b) Formulate a Picard iteration method at the PDE level, using the most recently computed u function in the α (u) coefficient. Derive general formulas for the entries in the linear system to be solved in each Picard iteration. Use the solution at the previous time step as initial guess for the Picard iteration

Based on the prior derived variational form of nonlinear diffusion equation 8), we introduce the Picard iteration method to iteratively solve this PDE. We can linearize $\alpha(u)$ and $f(x,t)$ items by using the previously computed u value as the input. Let u- be the approximation from the previous iteration, then we can derive the linearization version of the diffusion equation:

$$\int_\Omega (uv + \frac{\Delta t}{\rho}.\alpha(u^-)\nabla u.\nabla v)dx = \int_\Omega (u^{-1}v + \frac{\Delta t}{\rho}f(u^-)v)dx \quad (9)$$

This is equal to a linear system, a(u,v) = L(v) as following,

$$a(u,v) = \int_\Omega (uv + \frac{\Delta t}{\rho}.\alpha(u^-)\nabla u.\nabla v)dx \quad (10)$$

$$L(v), = \int_\Omega (u^{-1}v + \frac{\Delta t}{\rho}f(u^-)v)dx \quad (11)$$

By using the initial guess u^{-1} as u^- and iteratively solving the linear equations, we can approximate the u by u-.

c) Restrict the Picard iteration to a single iteration. That is, simply use a u value from the previous time step in the α(u) coefficient. Implement this method with the aid of the FEniCS software (in a dimension-independent way such that the code runs in 1D, 2D, and 3D).

As shown from the derived linear system a(u,v) = L(v), in this case I will limit the Picard iteration to one, and use the initial guess u^{-1} as u^- both in functions $\alpha(u^-)$and $f(u^-)$ for the calculation. The following programs have been coded in python and set the iteration to 1 for Picard iteration.

d) The first verification of the FEniCS implementation may reproduce a constant solution. Find values of the input data ρ , α, f, and I such that u(x, t) = C, where C is some chosen constant. Write test functions that verify the computation of a constant solution in 1D, 2D, and 3D for P1 and P2 elements (use simple domains: interval, square, box).

In order to test the correctness of the implemented Picard PDE solving program, we can verify this method by using the constant solution.

In this case I have set the input with $\rho = 1, \alpha(u) = u, f = 0$ and $u(x,t) = 10$ for the verification. The corresponding verification tests for 1D, 2D, and 3D case with P1 and P2 elements have been implemented in the program and the results shows the implementation is correct.

e) The second verification of the FEniCS implementation may reproduce a simple analytical solution of the PDE problem. Assume a(u) = 1, f = 0, Ω= [0, 1] × [0, 1], P1 elements, and $I(x, y) = cos(\pi x)$. The exact solution is then$u(x, y, t) = e^{-\pi^2 t} cos(\pi x)$,.... Show that E/h remains approximately constant as the mesh in space and time is simultaneously refined.

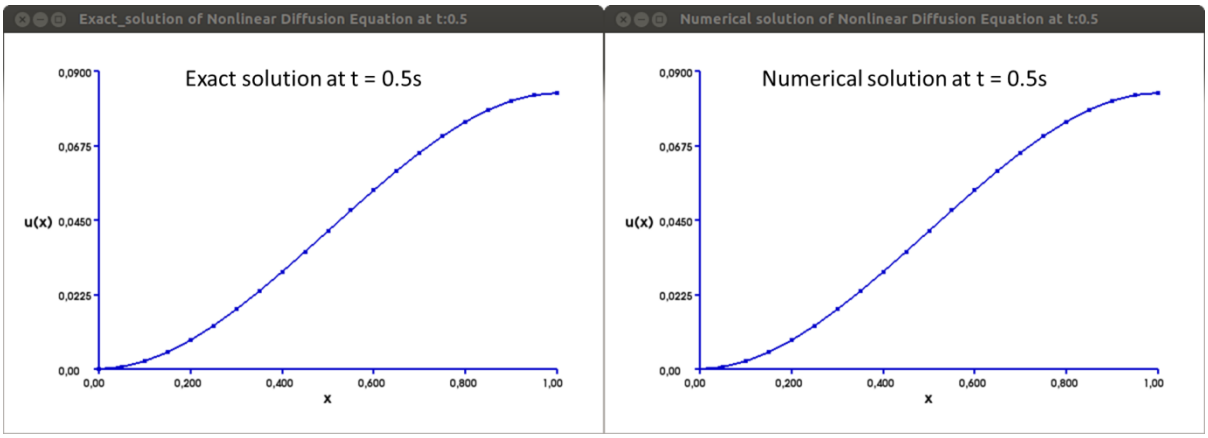
Based on the implemented backward Euler and Picard iteration method, we defined the following 5 tests with decreasing h parameters (h = dt = dx^2=dy^2), and the calculated E/h shows it approaching to a constant 0.0015 when the mesh in space and time is simultaneous refined.

	Preset Parameter h:	Calculated Error	Ratio of E/h
Test-1	h: 0.010000	E: 0.000020	E/h: 0.001950
Test-2	h: 0.005000	E: 0.000009	E/h: 0.001714
Test-3	h: 0.002500	E: 0.000004	E/h: 0.001576
Test-4	h: 0.001250	E: 0.000002	E/h: 0.001531
Test-5	h: 0.000625	E: 0.000001	E/h: 0.001516

f) The analytical solution in the test above is valid only for a linear version of the PDE without a source term f. To get an indication whether the implementation of the nonlinear diffusion PDE is correct or not, we can use the method of manufactured solutions. Say we restrict the problem to one space dimension, Ω= [0, 1], and choose following u(x,t) and a(u) = 1 + u2. Compare the FEniCS solution and the u given above as a function of x for a couple of t.

$$u(x,t) = t\int_0^x q(1-q)dq = tx^2\left(\frac{1}{2} - \frac{x}{3}\right)$$

In this test, we used the source term which derived from sympy, and made the comparison between the numerical results with the analytic results in the program. The results prove that the solved PDE by Picard method satisfied the exact solution. The following, I displayed the comparison of Exact solution and Numerical solution (t = 0.5s).



Picture 1. Comparison of Exact solution and Numerical solution (t = 0.5s).

g) List the different sources of numerical errors in the FEniCS program

The main error of FEnics program are the discretization error in time and space, if the discretization is fine enough, this error could be reduced. The other errors are derived from the iterations of Picard method and the error from the numerical integration, which are significantly smaller than the discretization errors.

h) To verify the nonlinear PDE implementation in FEniCS by checking convergence rate, we must eliminate the error due to a single Picard iteration. This can be done by finding a manufactured solution that fulfills the PDE with $u_{(1)}$, where $u_{(1)}$ is the solution at the previous time step. Perform a convergence rate test by decreasing h and observe that r -> 1.

In order to checking the convergence rate, we derived the following test schemes with decreasing h parameters, and based on the derived calculated Error, the program gives the convergence rate 1.02 in the end.

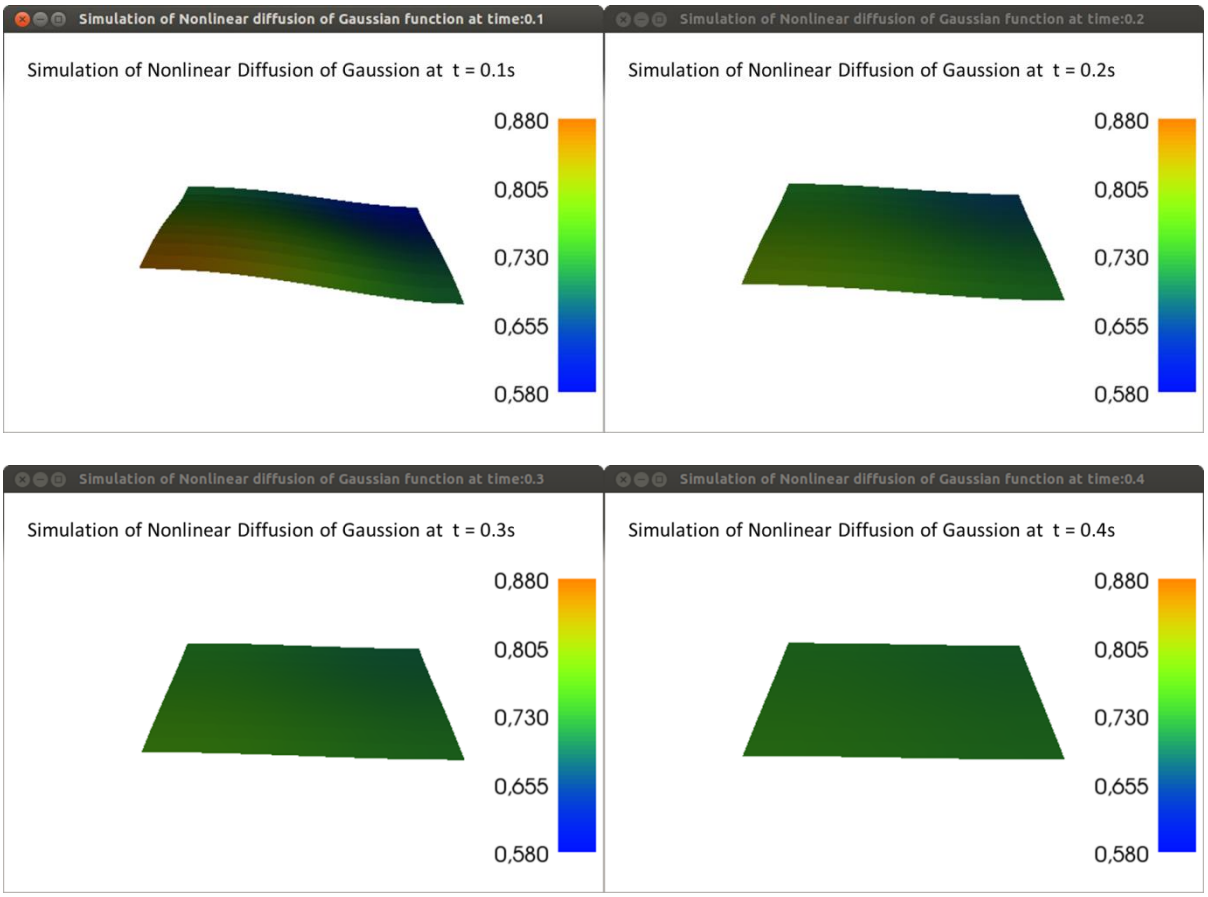
	Preset Parameter h:	Calculated Error
Test-1	h: 0.010000	1.18637397198e-05
Test-2	h: 0.005000	6.22647151689e-06
Test-3	h: 0.002500	3.13345474544e-06
Test-4	h: 0.001250	1.60736153672e-06
Test-5	h: 0.000625	7.9093607557e-07
Test-6	h: 0.0003125	3.90223603131e-07

i) Simulate the nonlinear diffusion of Gaussian function.

$$I(\boldsymbol{x},t) = \exp\left(\frac{1}{-2\sigma^2}(x^2+y^2)\right), (x,y)\epsilon\Omega = [0,1]x[0,1]\ and\ \frac{\vartheta u}{\vartheta n} = 0$$

To simulate the nonlinear diffusion of Gaussian function, we used the above initial condition with the selected parameters $\sigma = 1.0\ and\ \beta = 1.0$ with the chosen source term.

We simulated this nonlinear diffusion Gaussian function from 0 to 1s, and the following pictures display the simulation result from t=0.1s to t = 0.4s.



Picture 2. Simulation of Nonlinear Diffusion Gaussian Function from t= 0.1s to t=0.4s