Homework 6

April 23, 2021

Contents

Problem 1 ($\mathbf{u}_t + A\mathbf{u}_x = \mathbf{0}$) **2**

Problem 2. **4**
State $i = 0$ . . . 6
State $i = 1$ . . . 6
State $i = 2$ . . . 6

Problem 3 ($-\alpha u_{xx} + Au = f$) **8**
Part A . . . 8
Fourth-order element . . . 8
Convergence . . . 9
Part B: Transient Analysis . . . 11
SDIRK Implementation . . . 12
Convergence . . . 14

Appendix **16**
Source Code . . . 16
Fourth-order Lagrange element . . . 16
Analytic Transient Solution . . . 18

Problem 1 ($\mathbf{u}_t + A\mathbf{u}_x = \mathbf{0}$)

Consider the Riemann problem

$$\begin{pmatrix} p \\ u \end{pmatrix}_t + \begin{pmatrix} 0 & c_0^2\rho_0 \\ 1/\rho_0 & 0 \end{pmatrix} \begin{pmatrix} p \\ u \end{pmatrix}_x = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \qquad (x \in \mathbb{R}\ ,\ t > 0)$$

$$p(x,0) = \left\{ \begin{array}{ll} p_L & x < 0 \\ p_R & x > 0 \end{array} \right\}, \quad u(x,0) = \left\{ \begin{array}{ll} u_L & x < 0 \\ u_R & x > 0 \end{array} \right\}$$

describing linear acoustics. Compute the exact solution $p(x,t)$ and $u(x,t)$ for $x \in \mathbb{R}$, $t > 0$. Here c_0 and ρ_0 are positive constants while u_L, u_R, p_L and p_R are arbitrary real constants.

$$A = Q\Lambda Q^{-1} = \begin{pmatrix} 1 & -c_0\rho_0 \\ \frac{1}{c_0\rho_0} & 1 \end{pmatrix} \begin{pmatrix} c_0 & 0 \\ 0 & -c_0 \end{pmatrix} \begin{pmatrix} 1/2 & c_0\rho_0/2 \\ \frac{-1}{2c_0\rho_0} & 1/2 \end{pmatrix}$$

$$\mathbf{w}(x,0) = \begin{cases} \mathbf{w}_L & x < 0 \\ \mathbf{w}_R & x > 0 \end{cases}$$

Characteristics:

$$x_i^0 = x - \lambda_i t \quad i = 1, 2$$

$$w_i(x,t) = \begin{cases} w_i^L & x_0(x,t) < 0 \\ w_i^R & x_0(x,t) > 0 \end{cases}$$

Using $\mathbf{w} = Q^{-1}\mathbf{u}$

$$w_1 = \frac{p + c_0\rho_0 u}{2}$$

$$w_2 = \frac{u}{2} - \frac{1}{2c_0\rho_0}p$$

where the superscript

Decomposing Q into column vectors $\mathbf{q}_j$, and summing on $j = 1, 2$:

$$\mathbf{u}(x,t) = \begin{cases} \mathbf{q}_j w_j^L & x_i^0(x,t) < 0 \\ \mathbf{q}_j w_j^R & x_i^0(x,t) > 0 \end{cases}$$

Expanding for all regions yields

$$\mathbf{u}(x,t) = \begin{cases} \mathbf{q}_1 w_1^L + \mathbf{q}_2 w_2^L & x_2^0(x,t) < 0 \\ \mathbf{q}_1 w_1^L + \mathbf{q}_2 w_2^R & x_1^0(x,t) > 0 > x_2^0(x,t) \\ \mathbf{q}_1 w_1^R + \mathbf{q}_2 w_2^L & x_1^0(x,t) < 0 < x_2^0(x,t) \\ \mathbf{q}_1 w_1^R + \mathbf{q}_2 w_2^R & x_1^0(x,t) > 0 \end{cases}$$

The region $x_1^0 > 0 > x_2^0$ is not valid for the eigenvalues $\lambda = (c_0, -c_0)^T$ so that the only intermediate state is

$$\mathbf{u}^M = \mathbf{q}_1 w_1^R + \mathbf{q}_2 w_2^L, \quad x_1^0(x,t) < 0 < x_2^0(x,t)$$

$$\boxed{\begin{gathered} p^M = \frac{1}{2}\left(p^R + c_0\rho_0 u^R\right) - \frac{1}{2}\left(c_0\rho_0 u^L - p^L\right) \\ u^M = \frac{1}{2\rho_0 c_0}\left(p^R + \rho_0 c_0 u^R\right) + \frac{u^R}{2} - \frac{p^L}{2\rho_0 c_0} \\ \mathbf{u}^M = \begin{pmatrix} p^M \\ u^M \end{pmatrix} \\ \mathbf{u}(x,t) = \begin{cases} \mathbf{u}^L & x_2^0(x,t) < 0 \\ \mathbf{u}^M & x_1^0(x,t) < 0 < x_2^0(x,t) \\ \mathbf{u}^R & x_1^0(x,t) > 0 \end{cases} \end{gathered}}$$

Problem 2.

Consider the nonlinear isothermal equations of gas dynamics,

$$\begin{pmatrix}\rho\\ m\end{pmatrix}_t + \begin{pmatrix} m \\ \frac{m^2}{\rho} + a^2\rho\end{pmatrix}_x = \begin{pmatrix}0\\0\end{pmatrix}.$$

Here m is the momentum and the velocity of the gas is $v = m/\rho$. Let $a = 1$ and consider the Reimann problem $\rho(x,0) = \left\{\begin{array}{cc}\rho_0 & x<0\\ \rho_2 & x>0\end{array}\right\}$, $m(x,0) = \left\{\begin{array}{cc}m_0 & x<0\\ m_2 & x>0\end{array}\right\}$, where

$$q_0 = \begin{pmatrix}\rho_0\\ m_0\end{pmatrix} = \begin{pmatrix}1\\3\end{pmatrix}, \qquad q_2 = \begin{pmatrix}\rho_2\\ m_2\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix}.$$

Find an intermediate state $q_1 = \begin{pmatrix}\rho_1\\ m_1\end{pmatrix}$ and shock speeds $\dot{s}_1$ and $\dot{s}_2$ such that

$$F(q_i) - F(q_{i-1}) = \dot{s}_i(q_i - q_{i-1}), \qquad (i = 1,2), \qquad \dot{s}_2 > \dot{s}_1.$$

Following LeVeque (1992) and beginning at the Rankine-Hugoniot condition, the following system of equations is obtained:

$$\begin{aligned}\tilde{m} - \hat{m} &= s(\tilde{\rho} - \hat{\rho})\\ (\tilde{m}^2/\tilde{\rho} + a^2\tilde{\rho}) - (\hat{m}^2/\hat{\rho} + a^2\hat{\rho}) &= s(\tilde{m} - \hat{m})\end{aligned}$$

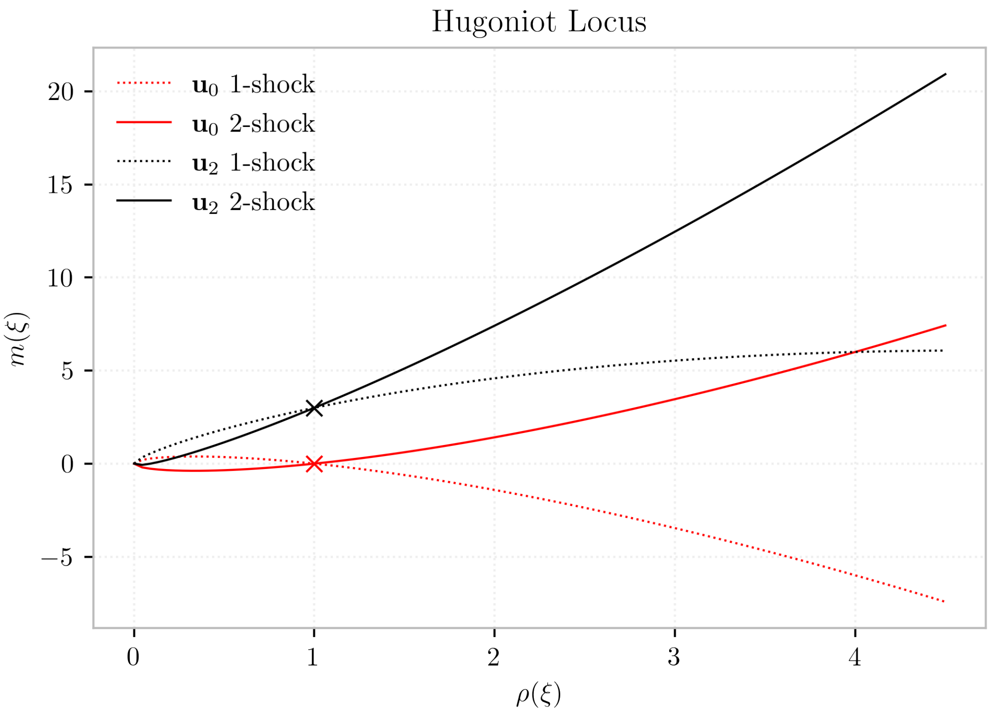

Figure 1: Graphical solution of the isothermal Riemann problem

$$\rho_1 m_0/\rho_0 - a\sqrt{\rho_1/\rho_0}\,(\rho_1 - \rho_0) = \rho_1 m_2/\rho_2 + a\sqrt{\rho_1/\rho_2}\,(\rho_1 - \rho_2)$$

$$\left(\frac{a}{\sqrt{\rho_2}} + \frac{a}{\sqrt{\rho_0}}\right) z^2 + \left(\frac{m_2}{\rho_2} - \frac{m_0}{\rho_0}\right) z - a\left(\sqrt{\rho_2} + \sqrt{\rho_0}\right) = 0$$

Plugging in the specified values for q_0 and q_2 yields the following coefficients:

$$\begin{aligned} \frac{a}{\sqrt{\rho_2}} + \frac{a}{\sqrt{\rho_0}} &= 2 \\ \frac{m_2}{\rho_2} - \frac{m_0}{\rho_0} &= -3 \\ a\left(\sqrt{\rho_2} + \sqrt{\rho_0}\right) &= -2 \end{aligned}$$

This yields the following roots:

$$\rho_1 = \left\{\frac{1}{4}, 4\right\}$$

$$m_1 = \rho_m m_r/\rho_r + a\sqrt{\rho_m/\rho_r}\,(\rho_m - \rho_r)$$

$$\boxed{q_1 = \begin{pmatrix} 4 \\ 6 \end{pmatrix}}$$

In each region $i = 0, 1, 2$, compute the characteristic speeds, which are the eigenvalues λ_{i1} and λ_{i2} of the Jacobian $J_i = DF(q_i)$. Also compute the fluid velocities v_i.

$$DF(q_i) = \begin{pmatrix} 0 & 1 \\ a^2 - m_i^2/\rho_i^2 & 2m_i/\rho_i \end{pmatrix}$$

State $i = 0$

$$DF(q_0) = \begin{pmatrix} 0 & 1 \\ -8 & 6 \end{pmatrix}$$

$$\lambda = \{2, 4\}$$

$$\mathbf{Q} = \begin{pmatrix} -0.4472136 & -0.24253563 \\ -0.89442719 & -0.9701425 \end{pmatrix}$$

$$v_0 = 3$$

State $i = 1$

$$\lambda_1 = \{1/2, 5/2\}$$

$$\mathbf{Q} = \begin{pmatrix} -0.89442719 & -0.37139068 \\ -0.4472136 & -0.92847669 \end{pmatrix}$$

$$v_1 = 3/2$$

$$\dot{s}_1 = 1$$

State $i = 2$

$$DF(q_2) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$\lambda_2 = (1, -1)$$

$$\mathbf{Q} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$$

$$v_2 = 0$$

$$\dot{s}_2 = 2$$

Confirm that $\lambda_{01} > \dot{s}_1 > \lambda_{11}$ and $\lambda_{12} > \dot{s}_2 > \lambda_{22}$, which are Lax's entropy conditions for this system. Also confirm that $v_0 > \dot{s}_1$, $v_1 > \dot{s}_1$, $\dot{s}_2 > v_1$ and $\dot{s}_2 > v_2$, which means fluid particles move from the original states q_0 and q_2 to the auxiliary state q_1 as the shocks propagate through space and time.

Problem 3 ($-\alpha u_{xx} + Au = f$)

Write a 1d finite element code to solve the modified Poisson equation

$$-\alpha u_{xx} + Au = f, \quad (0 \leq x \leq 1), \qquad u(0) = 0,\ u(1) = 0,$$

where $\alpha > 0$ and $A \geq 0$. Use 4th order finite elments on a uniformly spaced grid,

$$x_j = j/M, \qquad 0 \leq j \leq M$$

where M is divisible by 4 and the rth element includes nodes x_{4r+i} for $0 \leq r < M/4$ and $0 \leq i \leq 4$.

Part A

Multiplying by a test function v and integrating over the domain (applying integration by parts) yields:

$$\int_\Omega -\alpha u_{xx} v + \gamma uv = \int_\Omega fv$$

$$\int \alpha u_x v_x dx - \alpha u_x v| + \gamma \int uvdx = \int fvdx$$

For test functions which vanish on the boundary one obtains:

$$\int_\Omega \alpha u_x v_x + \gamma uv = \int_\Omega fv$$

$$a(\alpha u, v) + \langle Au, v\rangle = \langle f, v\rangle$$

Fourth-order element

A fourth order isoparametric 1D element is developed by applying Lagrange interpolation over 5 equally spaced sampling points.

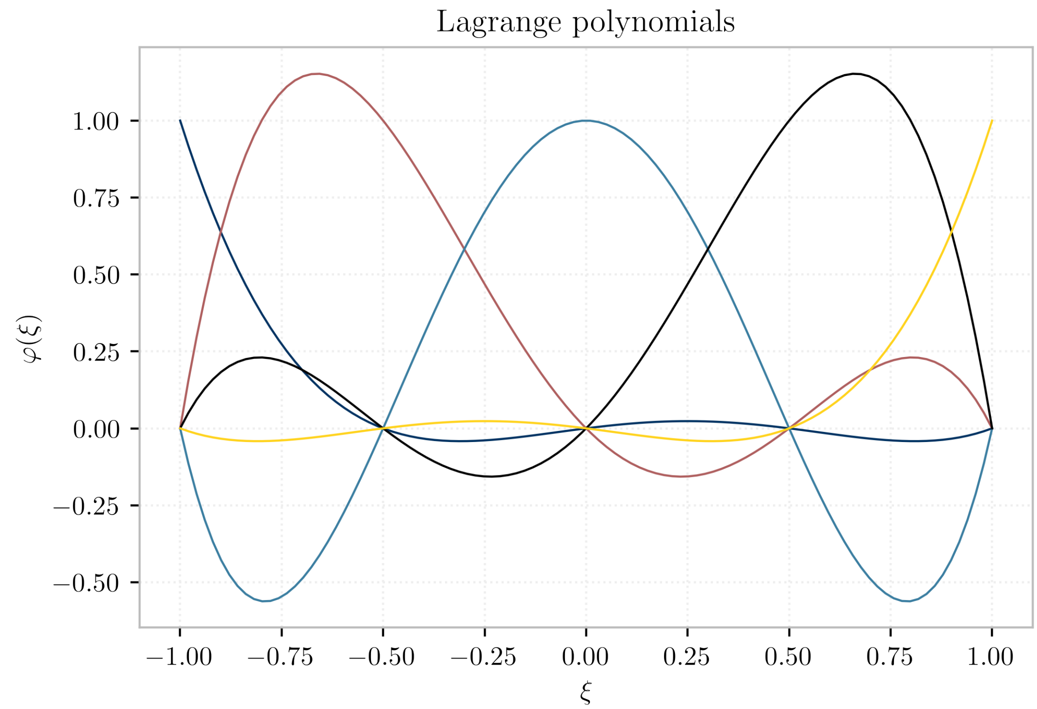

Figure 2: Shape functions

Convergence

(a) Do a convergence study for $\alpha = 1/100$, $A = 0$ and $f(x) = \frac{\pi^2}{100}\sum_{k=0}^{4} \sin\left((2k+1)\pi x\right)$.

The exact integral for this problem is as follows:

$$\frac{1}{\alpha 100}\sum \frac{1}{(2k+1)^2}\sin\left((2k+1)\pi x\right)$$

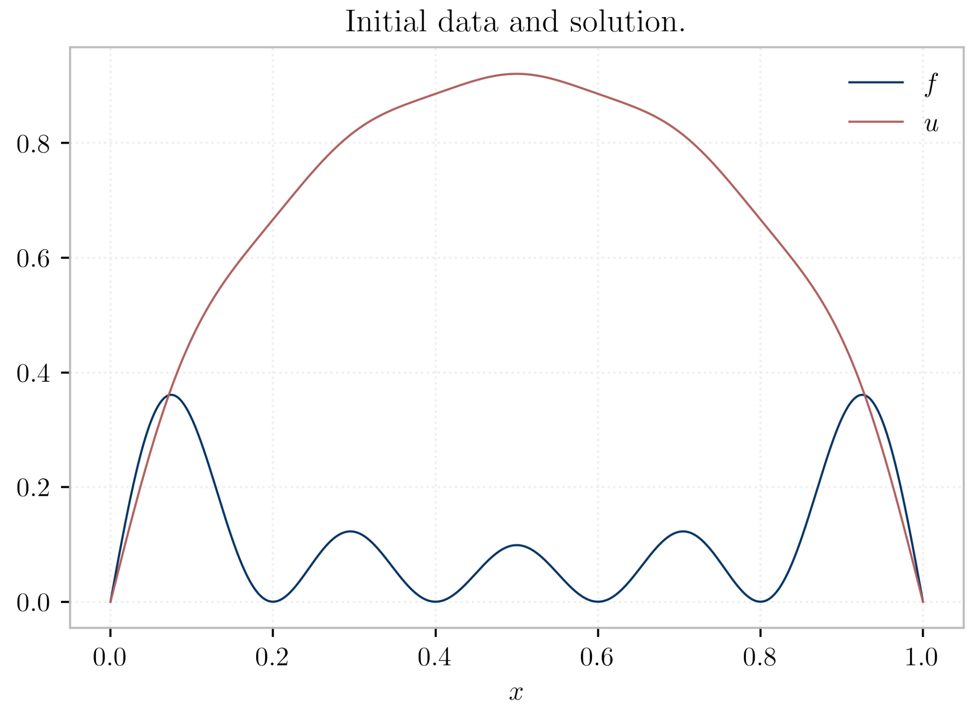

Figure 3: Source curve f and exact solution u

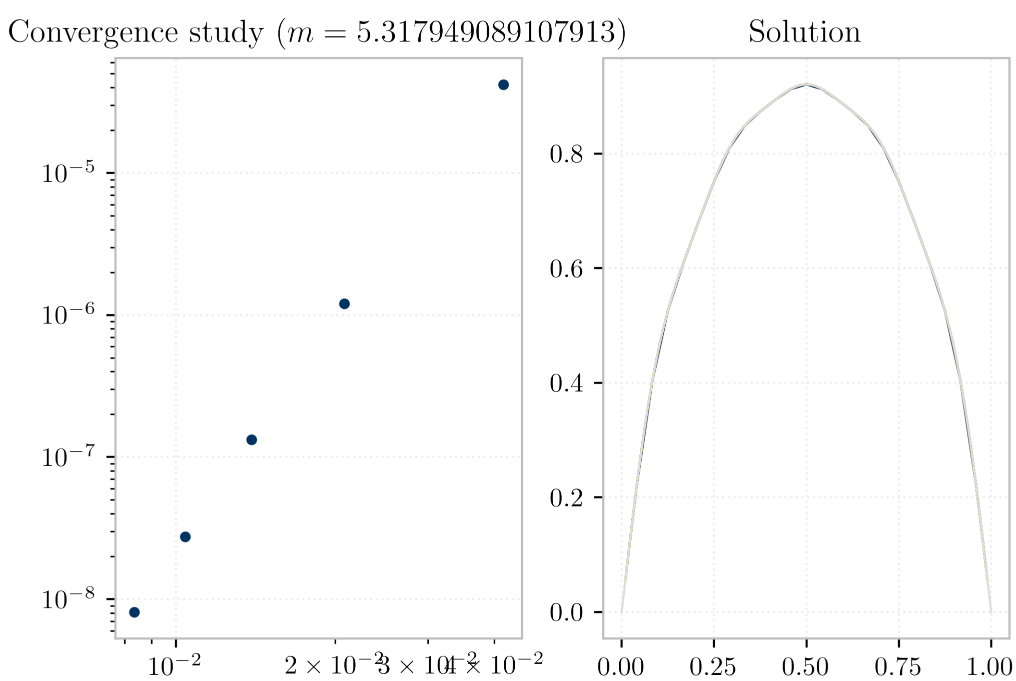

Figure 4: Convergence study for finite element solution of steady-state problem.

Part B: Transient Analysis

(b) Solve $u_t = \frac{1}{100}u_{xx} + f(x)\sin(\pi t)$ from $t = 0$ to $t = 1$ with initial condition $u(x, 0) = 0$ and the same f as in part (a). Use the 4th order implicit SDIRK timestepper with Butcher array given in problem 2 of HW 2. Use your finite element code above to solve the implicit equation for each stage of the timestepper. (I'll explain this in class). Make a convergence plot for several values of M and one choice of $\nu = k/h$ that you find works well.

The exact solution of the transient problem was derrived using the `sympy` CAS library. A plot is shown below for various times.

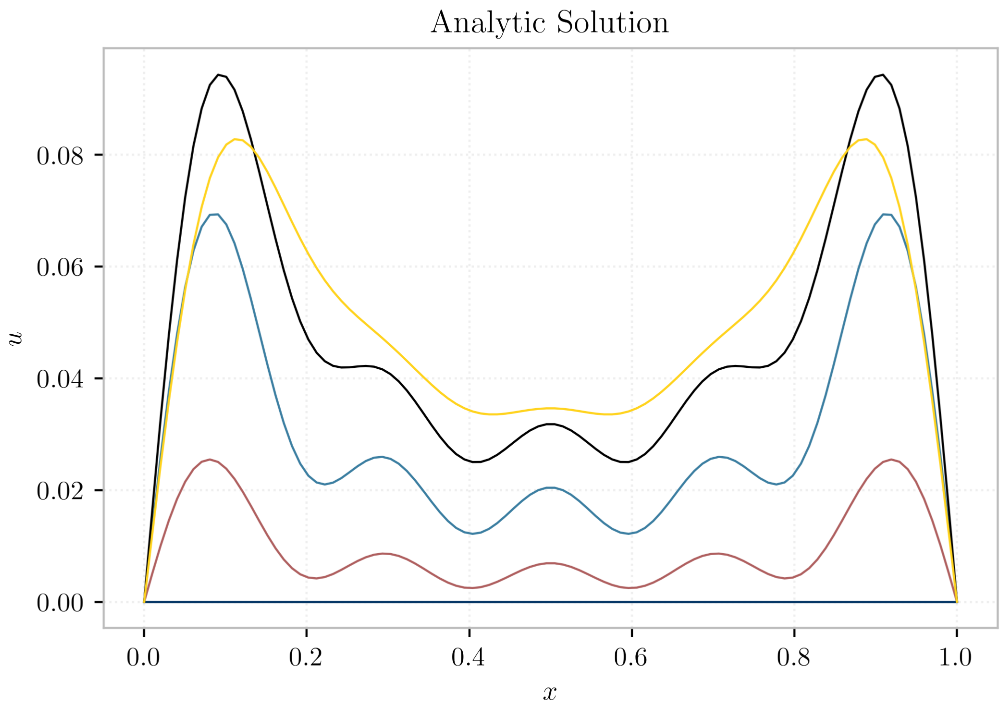

Figure 5: Analytic solution curves

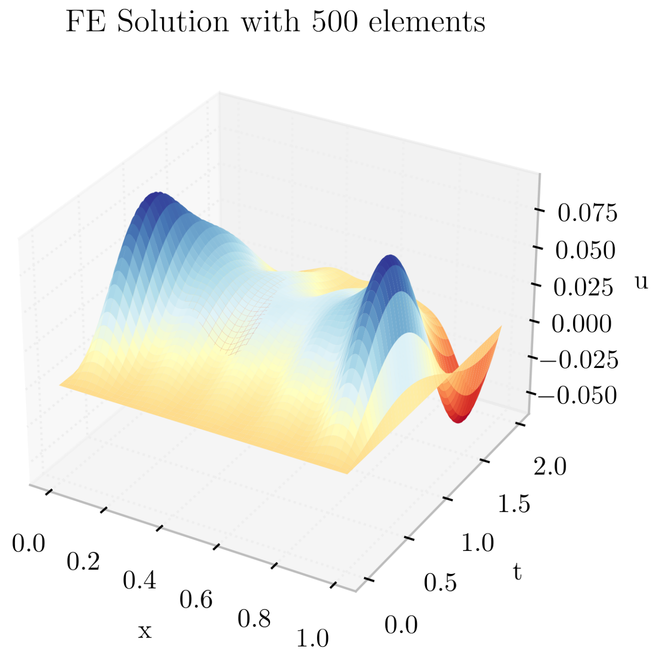

Figure 6: FEM solution

SDIRK Implementation

In HW2 a Runge-Kutta algorithm was implemented for problems with the following form:

$$\mathbf{u}_t + B\mathbf{u} = \mathbf{d}(t)$$

Where B is a linear operator . At stage i of a diagonal Runge-Kutta method one has

$$\ell_i = F\left(t_n + c_i k, \vec{u}_n + k\sum_{j=1}^{i-1} a_{ij}\ell_j + k a_{ij}\ell_i\right)$$

where $k = \Delta t$. For problems of the aforementioned form, this simplifies to

$$(I - k a_{ii} B)\,\ell_i = B\left(\vec{u}_n + k\sum_{j=1}^{i-1} a_{ij}\ell_j\right) + d(t_n + c_i k)$$

The following data is required to set up a particular scheme for $\mathbf{u} \in \mathbb{R}^n$ with $s \in \mathbf{Z}^+$ stages:

$\mathcal{T}$ A Butcher tableau with zero entries above the diagonal.

$B, \mathbb{R}^n \to \mathbb{R}^n$: Discrete space operator

$\mathbf{d}, \mathbb{R} \to \mathbb{R}^n$ Source term.

The problem at hand is manipulated to fit the following form:

$$\vec{u}_t = -\frac{1}{100} M^{-1} A\vec{u} + \vec{f}\sin\pi t$$

so that

$$B = \frac{-1}{100} M^{-1} A$$
$$\mathbf{d} = M^{-1} b \sin\pi t$$

where A, M, and b are the stiffness, mass and load vectors as readily produced by the implementation for Part B.

$$\left(M + \frac{k a_{ii}}{100} A\right) l_i = -\frac{1}{100} A\left(\vec{u}_n + k\sum_{j=1}^{i-1} a_{ij}\ell_j\right) + M\vec{f}\sin\pi\left(t_n + c_i k\right)$$

The tableau $\mathcal{T}$ is given as

$$\begin{array}{c|ccccc}
1/4 & 1/4 & & & & \\
3/4 & 1/2 & 1/4 & & & \\
11/20 & 17/50 & -1/25 & 1/4 & & \\
1/2 & 371/1360 & -137/2720 & 15/544 & 1/4 & \\
1 & 25/24 & -49/48 & 125/16 & -85/12 & 1/4 \\
\hline
 & 25/24 & -49/48 & 125/16 & -85/12 & 1/4
\end{array}$$

$u(x,t) = \sum_j u_j(t)\phi_j(x)$

$$u_t = \frac{1}{100} u_{xx} + f(x)\sin\pi t$$

let $u_t = \sum_j u_{j,t}\phi_j$, and $v = \phi_i$

$$\langle u_t, v\rangle = -\frac{1}{100} a(u,v) + \langle f, v\rangle \sin\pi t$$

$$M\vec{u}_t = -\frac{1}{100} A\vec{u} + M\vec{f}\sin\pi t$$

$$\vec{u}_t = -\frac{1}{100} M^{-1} A\vec{u} + \vec{f}\sin\pi t$$

Convergence

Convergence studies are presented below for time discretizations using both the SDIRK scheme provided and the Crank-Nicolson scheme.

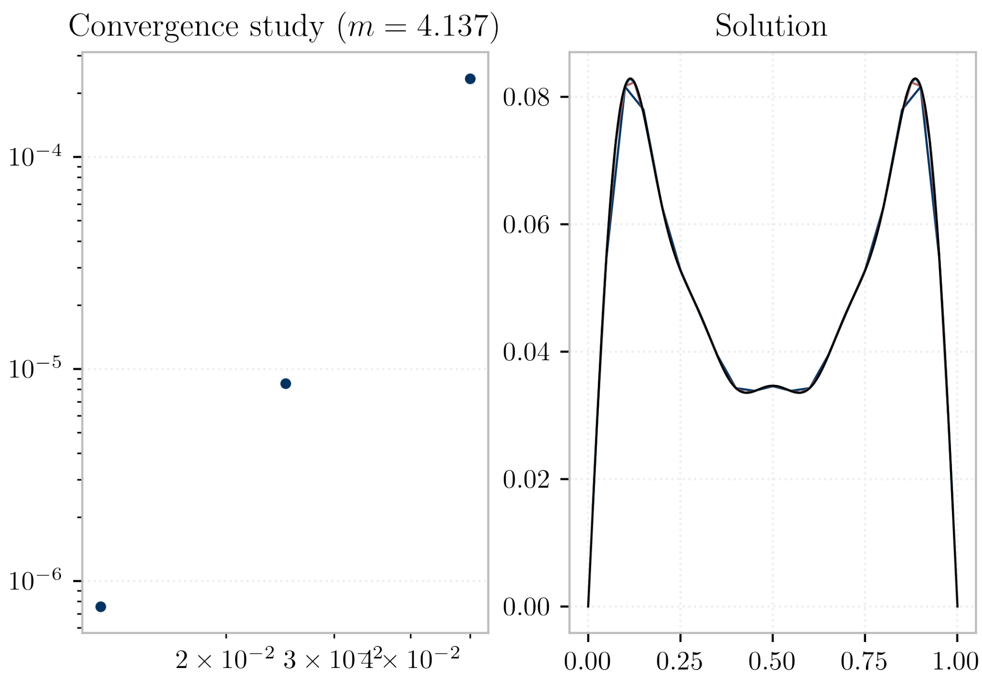

Figure 7: Convergence study.

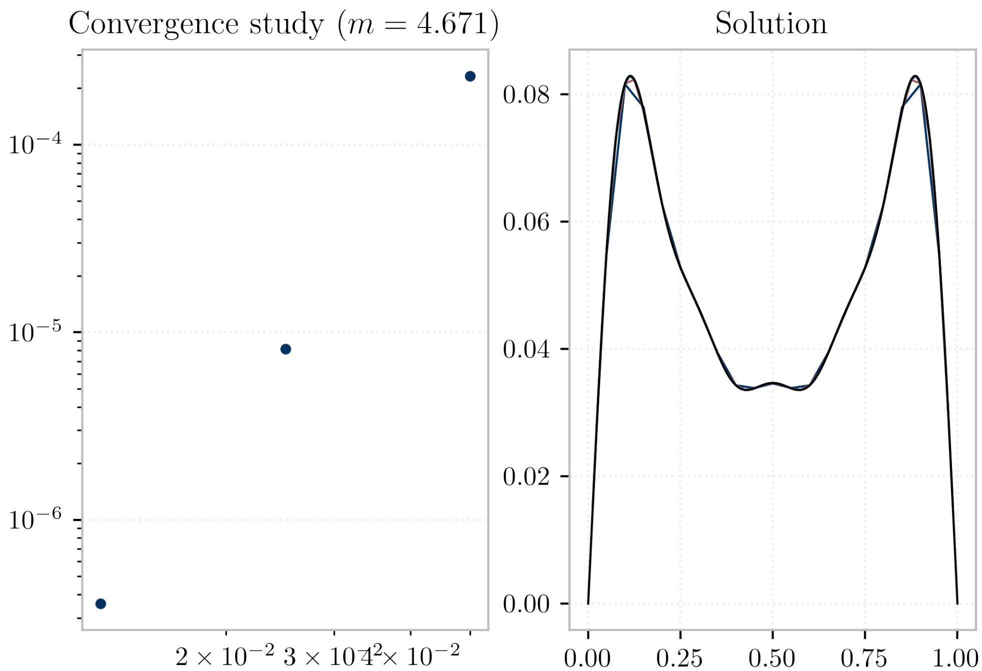

Figure 8: Convergence study for Crank-Nicolson tableau.

Appendix

Key source code excerpts are provided below. The Python libraries `elle`, `emme`, `anon` and `m228` which have been used throughout are self written and available either from PyPi.org via `pip` or on Github.

Source Code

Fourth-order Lagrange element

```
# external imports
import jax
# internal imports
import anon
import anon.atom as anp
from anon import quad

@anon.dual.generator(5)
def elem_0001(f=None,a1=1.0, a2=0.0,order=4):
    """
    Fourth order 1D Lagrange finite element with uniformly spaced nodes.

    Parameters
    ----------
    f: Callable
        element loading.
    a1: float
        Stiffness coefficient
    a2: float
        Mass coefficient
    """
    state = {}
    if f is None: f = lambda x: 0.0

    def transf(xi: float,x_nodes)->float:
        return ( x_nodes[0]*( + xi/6)
               + x_nodes[1]*( - 4*xi/3)
               + x_nodes[2]*( + 1)
               + x_nodes[3]*( + 4*xi/3)
               + x_nodes[4]*( - xi/6)
        )
    def grad_transf(xi,x_nodes):
        return abs(x_nodes[-1] - x_nodes[0])/2

    quad_points = quad.quad_points(n=order+1,rule="gauss-legendre")

    @jax.jit
    def jacx(u=None,y=None,state=None,xyz=None, a1=a1, a2=a2):
        x_nodes = anp.linspace(xyz[0][0],xyz[-1][0],5)
```

```
    grad = grad_transf(0,x_nodes)
    return a1*anp.array([
        [985/378, -3424/945, 508/315, -736/945, 347/1890],
        [-3424/945, 1664/189, -2368/315, 2944/945, -736/945],
        [508/315, -2368/315, 248/21, -2368/315, 508/315],
        [-736/945, 2944/945, -2368/315, 1664/189, -3424/945],
        [347/1890, -736/945, 508/315, -3424/945, 985/378],
    ]) / grad + a2*anp.array([
        [292/2835, 296/2835, -58/945, 8/405, -29/2835],
        [296/2835, 256/405, -128/945, 256/2835, 8/405],
        [-58/945, -128/945, 208/315, -128/945, -58/945],
        [8/405, 256/2835, -128/945, 256/405, 296/2835],
        [-29/2835, 8/405, -58/945, 296/2835, 292/2835],
    ])*grad


@jax.jit
def main(u,_,state,xyz,a1=a1,a2=a2):
    x_nodes = anp.linspace(xyz[0][0],xyz[-1][0],5)
    external_term = sum(
            anp.array([
              [f(transf(xi,x_nodes))*(2*xi**4/3 - 2*xi**3/3 - xi**2/6 + xi/6)],
              [f(transf(xi,x_nodes))*(-8*xi**4/3 + 4*xi**3/3 + 8*xi**2/3 - 4*xi/3)],
              [f(transf(xi,x_nodes))*(4*xi**4 - 5*xi**2 + 1)],
              [f(transf(xi,x_nodes))*(-8*xi**4/3 - 4*xi**3/3 + 8*xi**2/3 + 4*xi/3)],
              [f(transf(xi,x_nodes))*(2*xi**4/3 + 2*xi**3/3 - xi**2/6 - xi/6)],
            ]
        )*weight * grad_transf(xi,x_nodes) for xi, weight in zip(*quad_points)
    )
    resp = jacx(u,_,state,xyz,a1=a1,a2=a2)@u + external_term
    return u, resp, state
return locals()
```

Analytic Transient Solution

```
pi = anp.pi
sin = anp.sin
cos = anp.cos
exp = anp.exp
def cn(t):
    return [
        pi**2*(pi*alpha*sin(pi*t)/(pi**3*alpha**2 + pi)
            - cos(pi*t)/(pi**3*alpha**2 + pi))/100
            + pi**2/(100*(pi**3*alpha**2*exp(pi**2*alpha*t)
            + pi*exp(pi**2*alpha*t))),
        pi**2*(9*pi*alpha*sin(pi*t)/(81*pi**3*alpha**2 + pi)
            - cos(pi*t)/(81*pi**3*alpha**2 + pi))/100
            + pi**2/(100*(81*pi**3*alpha**2*exp(9*pi**2*alpha*t)
            + pi*exp(9*pi**2*alpha*t))),
        pi**2*(25*pi*alpha*sin(pi*t)/(625*pi**3*alpha**2 + pi)
            - cos(pi*t)/(625*pi**3*alpha**2 + pi))/100
            + pi**2/(100*(625*pi**3*alpha**2*exp(25*pi**2*alpha*t)
            + pi*exp(25*pi**2*alpha*t))),
        pi**2*(49*pi*alpha*sin(pi*t)/(2401*pi**3*alpha**2 + pi)
            - cos(pi*t)/(2401*pi**3*alpha**2 + pi))/100
            + pi**2/(100*(2401*pi**3*alpha**2*exp(49*pi**2*alpha*t)
            + pi*exp(49*pi**2*alpha*t))),
        pi**2*(81*pi*alpha*sin(pi*t)/(6561*pi**3*alpha**2 + pi)
            - cos(pi*t)/(6561*pi**3*alpha**2 + pi))/100
            + pi**2/(100*(6561*pi**3*alpha**2*exp(81*pi**2*alpha*t)
            + pi*exp(81*pi**2*alpha*t)))
     ]

def u(x,t):
    c = cn(t)
    return sum(
        c[n] * anp.sin(xi*x)
            for n,xi in enumerate([(2*k+1)*anp.pi for k in range(5)])
    )
```

preLeVeque, Randall J. 1992. *Numerical Methods for Conservation Laws.* Basel: Birkhäuser Basel. https://doi.org/10.1007/978-3-0348-8629-1.