Numerically Solving Partial Differential Equations
— Finite Difference Methods —

MECH 309 - *Numerical Methods in Mech. Eng.*

Prof. James Richard Forbes ©

McGill University, Department of Mechanical Engineering

Finite-Differences

- Consider $u(x,y,t)$.
- The forward-difference derivative approximation is

$$u_x = \frac{\partial u(x,y,t)}{\partial x} \approx \frac{u(x+\Delta x,y,t)-u(x,y,t)}{\Delta x},$$
$$u_y = \frac{\partial u(x,y,t)}{\partial y} \approx \frac{u(x,y+\Delta y,t)-u(x,y,t)}{\Delta y},$$
$$u_t = \frac{\partial u(x,y,t)}{\partial t} \approx \frac{u(x,y,t+\Delta t)-u(x,y,t)}{\Delta t}.$$

- The central-difference derivative approximation is

$$u_x = \frac{\partial u(x,y,t)}{\partial x} \approx \frac{u(x+\Delta x,y,t)-u(x-\Delta x,y,t)}{2\Delta x},$$
$$u_y = \frac{\partial u(x,y,t)}{\partial y} \approx \frac{u(x,y+\Delta y,t)-u(x,y-\Delta y,t)}{2\Delta y},$$
$$u_t = \frac{\partial u(x,y,t)}{\partial t} \approx \frac{u(x,y,t+\Delta t)-u(x,y,t-\Delta t)}{2\Delta t}.$$

- The central-difference second derivative approximation

$$
\begin{aligned}
u_{xx} &= \frac{\partial^2 u(x,y,t)}{\partial x^2} \approx \frac{u(x+\Delta x,y,t) - 2u(x,y,t) + u(x-\Delta x,y,t)}{(\Delta x)^2} \\
u_{yy} &= \frac{\partial^2 u(x,y,t)}{\partial y^2} \approx \frac{u(x,y+\Delta y,t) - 2u(x,y,t) + u(x,y-\Delta y,t)}{(\Delta y)^2} \\
u_{tt} &= \frac{\partial^2 u(x,y,t)}{\partial t^2} \approx \frac{u(x,y,t+\Delta t) - 2u(x,y,t) + u(x,y,t-\Delta t)}{(\Delta t)^2}.
\end{aligned}
$$

- Use a finite-difference approximation to discretize the PDE in time and/or space.

Semidiscrete FD Methods Applied to the Diffusion Equation

- Consider the diffusion (heat) equation

$$u_t = \alpha^2 u_{xx} + f, \quad 0 < x < \ell, \quad t \geq 0, \tag{1}$$

  where $f = f(x,t)$.

  - $f = f(x,t)$ represents some sort of internal energy sources, such as conversion of electrical energy to heat via a resistor.

- The ICs are

$$u(x,0) = \phi(x), \quad 0 < x < \ell.$$

- The BCs are

$$u(0,t) = \mu_0, \quad u(\ell,t) = \mu_\ell, \quad t \geq 0.$$

- A semidiscrete method discretizes (1) in space but keeps time continuous.

Discretize in Space, Numerically Integrate in Time

- Discretize in space, meaning $x_\imath = \imath\Delta x$, $\Delta x = \frac{\ell}{n+1}$, $x_{\imath+1} = x_\imath + \Delta x$, $x_{\imath-1} = x_\imath - \Delta x$, $\imath = 0, 1, \ldots, n+1$.
- Let $q_\imath(t) = u(x_\imath, t)$.
- Using a central-difference second derivative approximation to approximate u_{xx}, (1) becomes

$$\begin{aligned} \dot{u}(x_\imath, t) &= \frac{\alpha^2}{(\Delta x)^2}\left(u(x_{\imath+1}, t) - 2u(x_\imath, t) + u(x_{\imath-1}, t)\right) + f(x_\imath, t), \quad \imath = 1, \ldots, n, \\ \dot{q}_\imath(t) &= \kappa\left(q_{\imath+1}(t) - 2q_\imath(t) + q_{\imath-1}(t)\right) + f_\imath(t), \quad \imath = 1, \ldots, n, \end{aligned} \tag{2}$$

  where $\kappa = \frac{\alpha^2}{(\Delta x)^2}$ and $f_\imath(t) = f(x_\imath, t)$.
- From the ICs, $q_\imath(0) = u(x_\imath, 0) = \phi(x_\imath)$, $i = 1, \ldots, n-1, n$.
- From the BCs, $q_0(t) = u(0, t) = \mu_0$ and $q_{n+1}(t) = u(x_{n+1}, t) = u(\ell, t) = \mu_\ell$.

- For example, say $n = 4$.
- The BCs $q_0(t) = \mu_0$ and $q_{n+1}(t) = q_5(t) = \mu_\ell$.
- Writing out (2) explicitly gives

$$\begin{aligned}
\dot{q}_1(t) &= \kappa\left(q_2(t) - 2q_1(t) + \mu_0\right) + f_1(t),\\
\dot{q}_2(t) &= \kappa\left(q_3(t) - 2q_2(t) + q_1(t)\right) + f_2(t),\\
\dot{q}_3(t) &= \kappa\left(q_4(t) - 2q_3(t) + q_2(t)\right) + f_3(t),\\
\dot{q}_4(t) &= \kappa\left(\mu_\ell - 2q_4(t) + q_3(t)\right) + f_4(t),
\end{aligned}$$

where $q_\imath(0) = \phi(x_\imath)$, $i = 1, 2, 3, 4$.

- Writing out these four equations in first-order state-space form gives

$$\begin{bmatrix}\dot{q}_1(t)\\ \dot{q}_2(t)\\ \dot{q}_3(t)\\ \dot{q}_4(t)\end{bmatrix} = \underbrace{\kappa\begin{bmatrix}-2 & 1 & 0 & 0\\ 1 & -2 & 1 & 0\\ 0 & 1 & -2 & 1\\ 0 & 0 & 1 & -2\end{bmatrix}}_{\tilde{\mathbf{A}}}\underbrace{\begin{bmatrix}q_1(t)\\ q_2(t)\\ q_3(t)\\ q_4(t)\end{bmatrix}}_{\mathbf{q}(t)} + \begin{bmatrix}\kappa\mu_0\\ 0\\ 0\\ \kappa\mu_\ell\end{bmatrix} + \begin{bmatrix}f_1(t)\\ f_2(t)\\ f_3(t)\\ f_4(t)\end{bmatrix}, \qquad \begin{bmatrix}q_1(0)\\ q_2(0)\\ q_3(0)\\ q_4(0)\end{bmatrix} = \begin{bmatrix}\phi(x_1)\\ \phi(x_2)\\ \phi(x_3)\\ \phi(x_4)\end{bmatrix}, \quad (3)$$

$$\dot{\mathbf{q}}(t) = \mathbf{f}(\mathbf{q}(t), t), \qquad q_\imath(0) = \phi(x_\imath), \quad i = 1, 2, 3, 4,$$

which can solved using your favourite numerical integration method for ODEs!

  - For example, any Runge-Kutta method.

- Generalization to any n is straightforward.
- As before, the BCs $q_0(t) = \mu_0$ and $q_{n+1}(t) = \mu_\ell$.
- Writing out (2) explicitly gives

$$
\begin{aligned}
\dot{q}_1(t) &= \kappa\left(q_2(t) - 2q_1(t) + \mu_0\right) + f_1(t),\\
\dot{q}_2(t) &= \kappa\left(q_3(t) - 2q_2(t) + q_1(t)\right) + f_2(t),\\
\dot{q}_3(t) &= \kappa\left(q_4(t) - 2q_3(t) + q_2(t)\right) + f_3(t),\\
&\ \vdots\\
\dot{q}_{n-1}(t) &= \kappa\left(q_n(t) - 2q_{n-1}(t) + q_{n-2}(t)\right) + f_{n-1}(t),\\
\dot{q}_n(t) &= \kappa\left(\mu_\ell - 2q_n(t) + q_{n-1}(t)\right) + f_n(t),
\end{aligned}
$$

where $q_\imath(0) = \phi(x_\imath)$, $\imath = 1, \ldots, n$.

- Writing out these n equations in first-order state-space form gives

$$\begin{bmatrix} \dot{q}_1(t) \\ \dot{q}_2(t) \\ \dot{q}_3(t) \\ \vdots \\ \dot{q}_{n-2}(t) \\ \dot{q}_{n-1}(t) \\ \dot{q}_n(t) \end{bmatrix} = \kappa \underbrace{\begin{bmatrix} -2 & 1 & 0 & \cdots & 0 & 0 \\ 1 & -2 & 1 & \cdots & 0 & 0 \\ 0 & 1 & -2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -2 & 1 \\ 0 & 0 & \cdots & 0 & 1 & -2 \end{bmatrix}}_{\tilde{\mathbf{A}}} \underbrace{\begin{bmatrix} q_1(t) \\ q_2(t) \\ q_3(t) \\ \vdots \\ q_{n-2}(t) \\ q_{n-1}(t) \\ q_n(t) \end{bmatrix}}_{\mathbf{q}(t)} + \begin{bmatrix} \kappa\mu_0 \\ 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \\ \kappa\mu_\ell \end{bmatrix} + \begin{bmatrix} f_1(t) \\ f_2(t) \\ f_3(t) \\ \vdots \\ \vdots \\ f_{n-1}(t) \\ f_n(t) \end{bmatrix}, \tag{4}$$

$$\dot{\mathbf{q}}(t) = \mathbf{f}(\mathbf{q}(t), t), \qquad q_\imath(0) = \phi(x_\imath) \quad i = 1, \cdots, n,$$

which can solved using your favourite numerical integration method for ODEs!

- This solution method is known as the **method of lines** (MOL). MOL is a *semi*discrete method because space is discretized and time is left continuous.

Example [1, 2]

- Consider the diffusion (heat) equation
$$u_t = \alpha^2 u_{xx}, \quad 0 \le x \le 1, \quad t \ge 0, \tag{5}$$
where $\alpha^2 = 1$.
- The ICs are
$$u(x,0) = 100, \quad 0 \le x \le 1.$$
- The BCs are
$$u(0,t) = 0, \quad u(1,t) = 0, \quad t \ge 0.$$
- Equation (5) has an exact solution,
$$u(x,t) = \frac{400}{\pi} \sum_{k=1,3,5,\ldots}^{\infty} \frac{1}{k} \sin(k\pi x) e^{-(k\pi)^2 t},$$
that a numerical solution can be compared to.
- Let $t \in [0, 0.05]$ (s) with $\mathrm{d}t = 1 \times 10^{-4}$ (s).
- Consider $n = 10, 20, 50, 100$.

$n = 10$

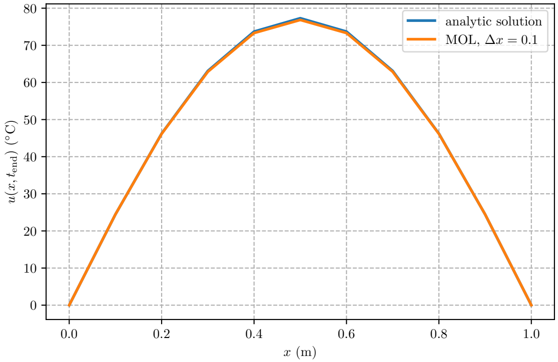

Figure 1: Response at t_{end}.

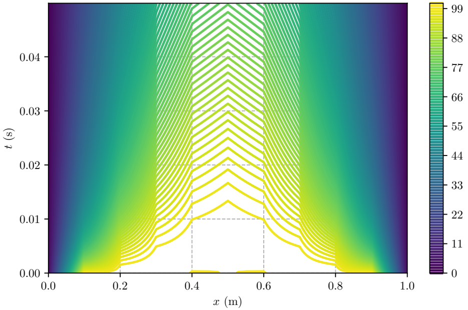

Figure 2: Response over time as a contour plot.

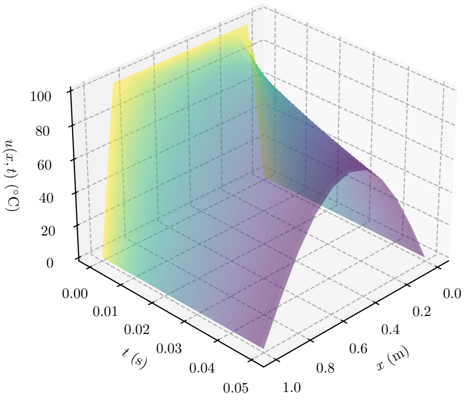

Figure 3: Response over time at a 3D plot.

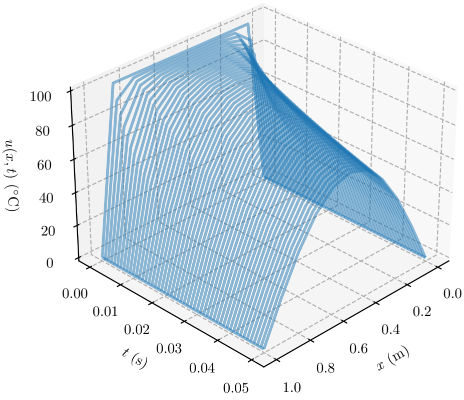

Figure 4: Wireframe response over time at a 3D plot.

$n = 20$

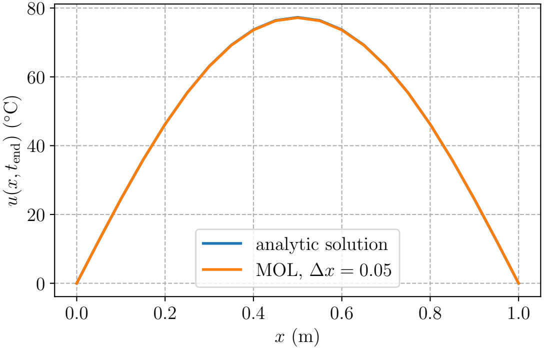

Figure 5: Response at t_{end}.

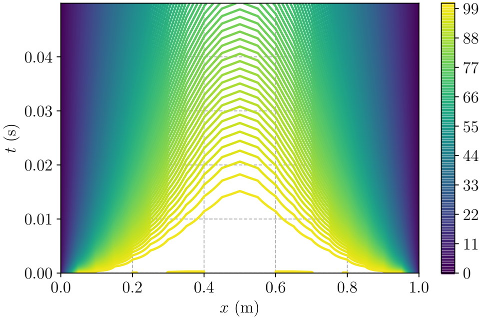

Figure 6: Response over time as a contour plot.

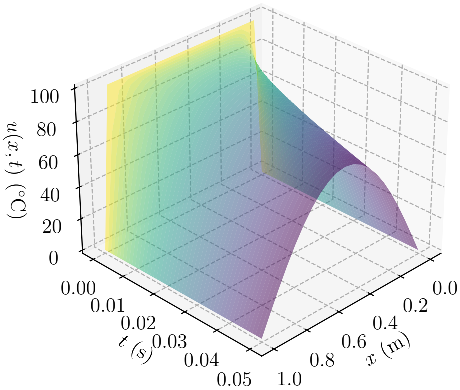

Figure 7: Response over time at a 3D plot.

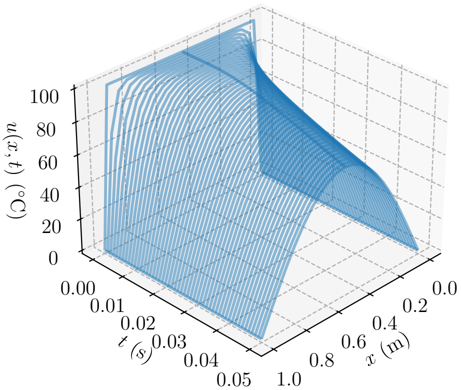

Figure 8: Wireframe response over time at a 3D plot.

$n = 50$

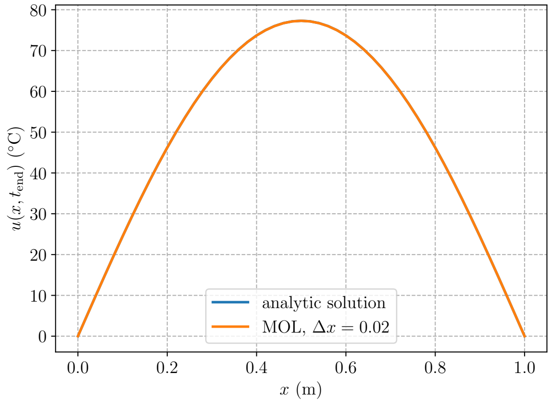

Figure 9: Response at t_{end}.

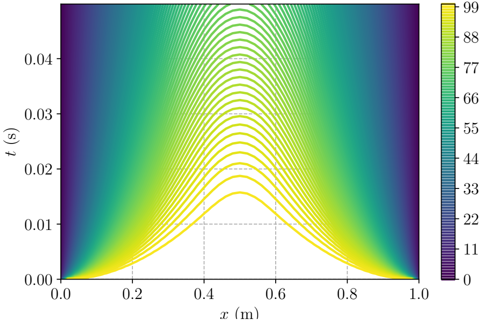

Figure 10: Response over time as a contour plot.

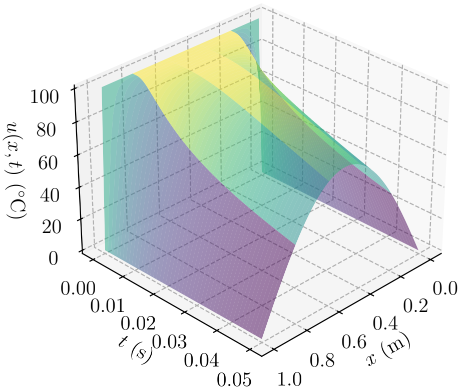

Figure 11: Response over time at a 3D plot.

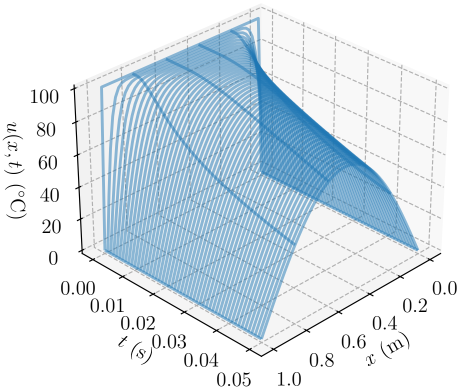

Figure 12: Wireframe response over time at a 3D plot.

$n = 100$

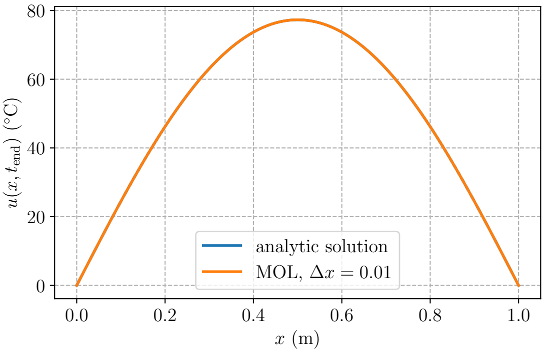

Figure 13: Response at t_{end}.

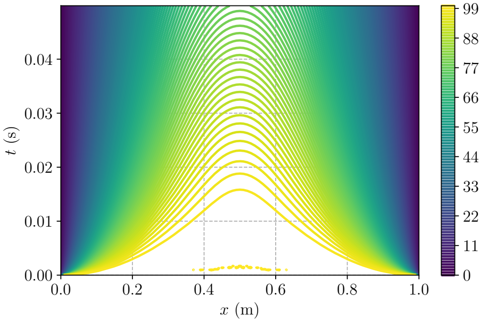

Figure 14: Response over time as a contour plot.

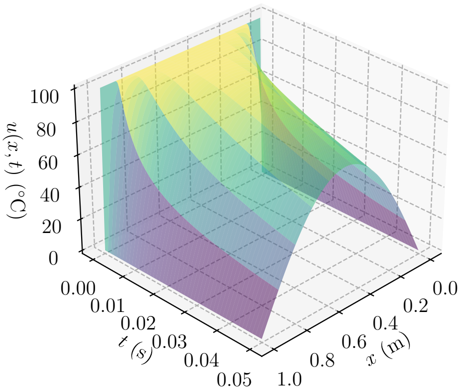

Figure 15: Response over time at a 3D plot.

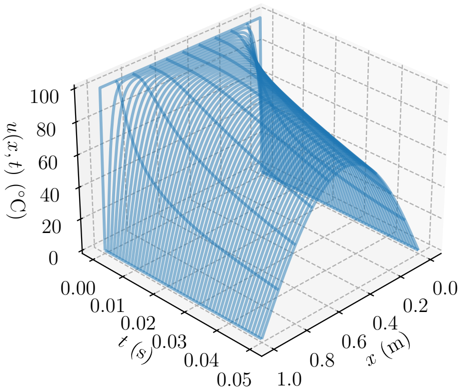

Figure 16: Wireframe response over time at a 3D plot.

Semidiscrete FD Methods Applied to the Wave Equation

- Consider the wave equation

$$u_{tt} = \alpha^2 u_{xx} + f, \quad 0 < x < \ell, \quad t \geq 0, \tag{6}$$

  where $f = f(x,t)$.

  - $f = f(x,t)$ represents some sort of external loads applied to a flexible mechanical system.

- The ICs are

$$\begin{aligned} u(x,0) &= \phi(x), \quad 0 < x < \ell, \\ \dot{u}(x,0) &= \psi(x), \quad 0 < x < \ell. \end{aligned}$$

- The BCs are

$$u(0,t) = \mu_0, \quad u(\ell,t) = \mu_\ell, \quad t \geq 0.$$

- A semidiscrete method discretizes (6) in space but keeps time continuous.

Discretize in Space, Numerically Integrate in Time

- Discretize in space, meaning $x_\imath = \imath \Delta x$, $\Delta x = \frac{\ell}{n+1}$, $x_{\imath+1} = x_\imath + \Delta x$, $x_{\imath-1} = x_\imath - \Delta x$, $\imath = 0, 1, \ldots, n+1$.
- Let $q_\imath(t) = u(x_\imath, t)$.
- Using a central-difference second derivative approximation to approximate u_{xx}, (6) becomes

$$\ddot{u}(x_\imath, t) = \frac{\alpha^2}{(\Delta x)^2} \left(u(x_{\imath+1}, t) - 2u(x_\imath, t) + u(x_{\imath-1}, t)\right) + f(x_\imath, t), \quad \imath = 1, \ldots, n,$$

$$\ddot{q}_\imath(t) = \kappa \left(q_{\imath+1}(t) - 2q_\imath(t) + q_{\imath-1}(t)\right) + f_\imath(t), \quad \imath = 1, \ldots, n, \tag{7}$$

where $\kappa = \frac{\alpha^2}{(\Delta x)^2}$ and $f_\imath(t) = f(x_\imath, t)$

- From the ICs, $q_\imath(0) = u(x_\imath, 0) = \phi(x_\imath)$, $\dot{q}_\imath(0) = \dot{u}(x_\imath, 0) = \psi(x_\imath)$, $i = 1, \ldots, n-1, n$.
- From the BCs, $q_0(t) = u(0, t) = \mu_0$ and $q_{n+1}(t) = u(x_{n+1}, t) = u(\ell, t) = \mu_\ell$.
- Note that, along with (7), there are "velocity" states, meaning

$$\dot{q}_\imath(t) = v_\imath(t), \tag{8}$$

$$\dot{v}_\imath(t) = \kappa \left(q_{\imath+1}(t) - 2q_\imath(t) + q_{\imath-1}(t)\right) + f_\imath(t), \qquad (7)$$

must be numerically integrated simultaneously.

- For example, say $n = 4$.
- The BCs $q_0(t) = \mu_0$ and $q_{n+1}(t) = \mu_\ell$.
- Writing out (7) and (8) explicitly gives

$$
\begin{aligned}
\dot{q}_1(t) &= v_1(t), \\
\dot{q}_2(t) &= v_2(t), \\
\dot{q}_3(t) &= v_3(t), \\
\dot{q}_4(t) &= v_4(t), \\
\dot{v}_1(t) &= \kappa\left(q_2(t) - 2q_1(t) + \mu_0\right) + f_1(t), \\
\dot{v}_2(t) &= \kappa\left(q_3(t) - 2q_2(t) + q_1(t)\right) + f_2(t), \\
\dot{v}_3(t) &= \kappa\left(q_4(t) - 2q_3(t) + q_2(t)\right) + f_3(t), \\
\dot{v}_4(t) &= \kappa\left(\mu_\ell - 2q_4(t) + q_3(t)\right) + f_4(t),
\end{aligned}
$$

where $q_\imath(0) = \phi(x_\imath)$, $v_\imath(0) = \psi(x_\imath)$, $i = 1, 2, 3, 4$.

- Let

$$\mathbf{q}(t) = \begin{bmatrix} q_1(t) \\ q_2(t) \\ q_3(t) \\ q_4(t) \end{bmatrix}, \qquad \dot{\mathbf{q}}(t) = \begin{bmatrix} \dot{q}_1(t) \\ \dot{q}_2(t) \\ \dot{q}_3(t) \\ \dot{q}_4(t) \end{bmatrix} = \begin{bmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \\ v_4(t) \end{bmatrix} = \mathbf{v}(t), \qquad \tilde{\mathbf{b}}(t) = \begin{bmatrix} \kappa\mu_0 \\ 0 \\ 0 \\ \kappa\mu_\ell \end{bmatrix} + \begin{bmatrix} f_1(t) \\ f_2(t) \\ f_3(t) \\ f_4(t) \end{bmatrix}.$$

- Writing out these four equations in first-order state-space form gives

$$\underbrace{\begin{bmatrix} \dot{\mathbf{q}}(t) \\ \dot{\mathbf{v}}(t) \end{bmatrix}}_{\dot{\mathbf{x}}(t)} = \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{1} \\ \tilde{\mathbf{A}} & \mathbf{0} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \mathbf{q}(t) \\ \mathbf{v}(t) \end{bmatrix}}_{\mathbf{x}(t)} + \begin{bmatrix} \mathbf{0} \\ \tilde{\mathbf{b}}(t) \end{bmatrix}, \qquad \mathbf{x}(0) = \begin{bmatrix} \mathbf{q}(0) \\ \mathbf{v}(0) \end{bmatrix} \tag{9}$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), t), \qquad q_\imath(0) = \phi(x_\imath), \quad v_\imath(0) = \psi(x_\imath), \quad i = 1, 2, 3, 4,$$

where $\tilde{\mathbf{A}}$ is defined in (3) (or in (4) in the general case).

- Equation (9) can solved using your favourite numerical integration method for ODEs!
  - For example, any Runge-Kutta method.

- Now generalize. Let

$$\mathbf{q}(t) = \begin{bmatrix} q_1(t) \\ q_2(t) \\ q_3(t) \\ \vdots \\ q_{n-1}(t) \\ q_n(t) \end{bmatrix}, \qquad \dot{\mathbf{q}}(t) = \begin{bmatrix} \dot{q}_1(t) \\ \dot{q}_2(t) \\ \dot{q}_3(t) \\ \vdots \\ \dot{q}_{n-1}(t) \\ \dot{q}_n(t) \end{bmatrix} = \begin{bmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \\ \vdots \\ v_{n-1}(t) \\ v_n(t) \end{bmatrix} = \mathbf{v}(t), \qquad \tilde{\mathbf{b}}(t) = \begin{bmatrix} \kappa\mu_0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ \kappa\mu_\ell \end{bmatrix} + \begin{bmatrix} f_1(t) \\ f_2(t) \\ f_3(t) \\ \vdots \\ f_{n-1}(t) \\ f_n(t) \end{bmatrix}.$$

- The overall system to numerically integrate is then

$$\underbrace{\begin{bmatrix} \dot{\mathbf{q}}(t) \\ \dot{\mathbf{v}}(t) \end{bmatrix}}_{\dot{\mathbf{x}}(t)} = \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{1} \\ \tilde{\mathbf{A}} & \mathbf{0} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \mathbf{q}(t) \\ \mathbf{v}(t) \end{bmatrix}}_{\mathbf{x}(t)} + \begin{bmatrix} \mathbf{0} \\ \tilde{\mathbf{b}}(t) \end{bmatrix}, \qquad \mathbf{x}(0) = \begin{bmatrix} \mathbf{q}(0) \\ \mathbf{v}(0) \end{bmatrix} \tag{10}$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), t), \qquad q_\imath(0) = \phi(x_\imath), \quad v_\imath(0) = \psi(x_\imath), \quad i = 1, \cdots, n$$

where $\tilde{\mathbf{A}}$ is defined in (4).

- Equation (10) can solved using your favourite numerical integration method for ODEs!

Example [2]

- Consider the wave equation

$$u_{tt} = \alpha^2 u_{xx}, \quad 0 \leq x \leq 1, \quad t \geq 0, \tag{11}$$

  where $\alpha^2 = 1$.

- The ICs are

$$u(x,0) = \exp\left(-32(x-0.5)^2\right), \quad 0 \leq x \leq 1,$$
$$\dot{u}(x,0) = 0, \quad 0 \leq x \leq 1.$$

- The BCs are

$$u(0,t) = 0, \quad u(1,t) = 0, \quad t \geq 0.$$

- Let $t \in [0,2]$ (s) with $\mathrm{d}t = 1 \times 10^{-2}$ (s).
- Consider $n = 10, 20, 50, 100$.

$n = 10$

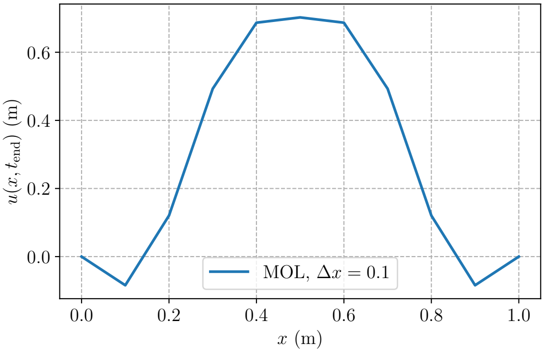

Figure 17: Response at t_{end}.

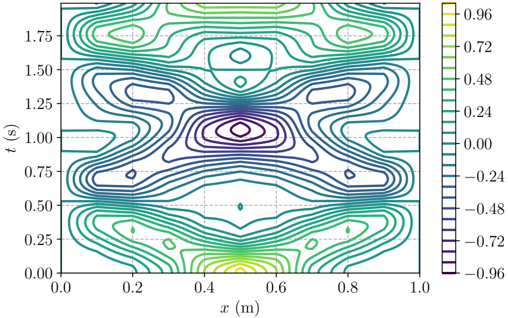

Figure 18: Response over time as a contour plot.

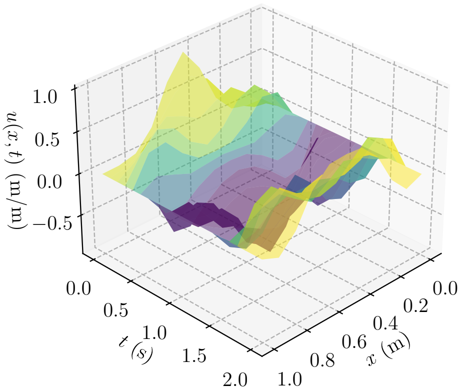

Figure 19: Response over time at a 3D plot.

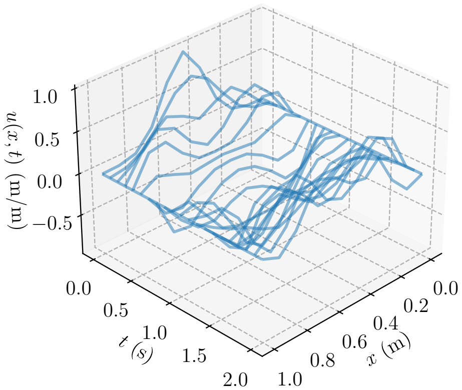

Figure 20: Wireframe response over time at a 3D plot.

$n = 20$

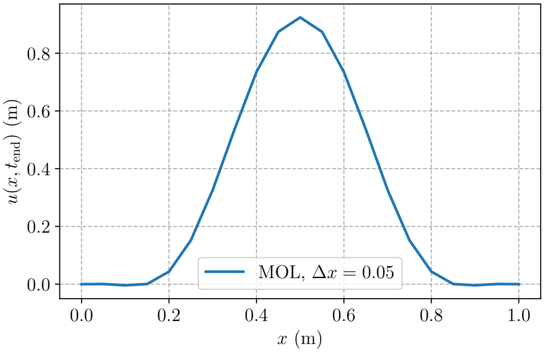

Figure 21: Response at t_{end}.

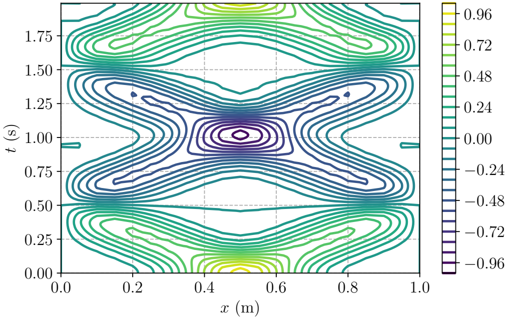

Figure 22: Response over time as a contour plot.

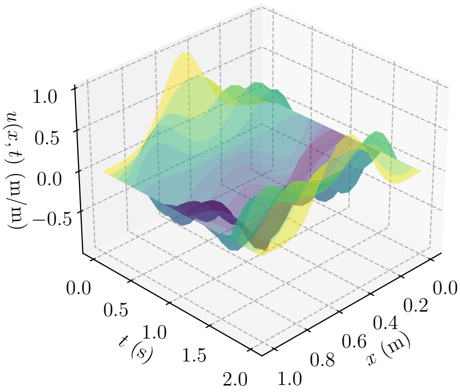

Figure 23: Response over time at a 3D plot.

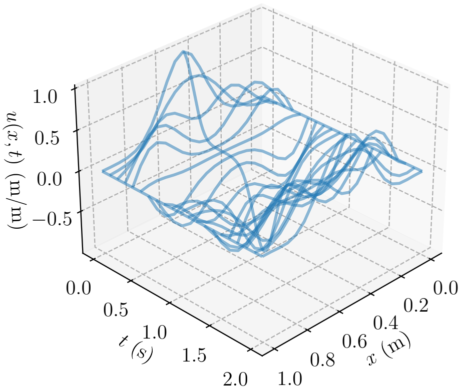

Figure 24: Wireframe response over time at a 3D plot.

$n = 50$

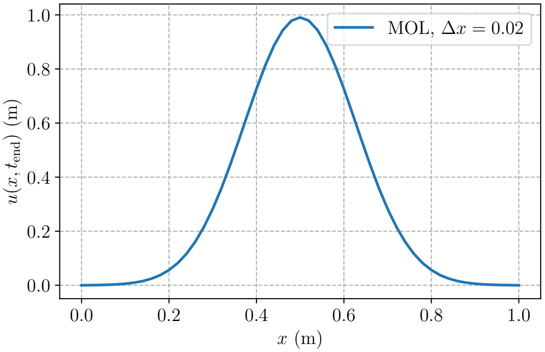

Figure 25: Response at t_{end}.

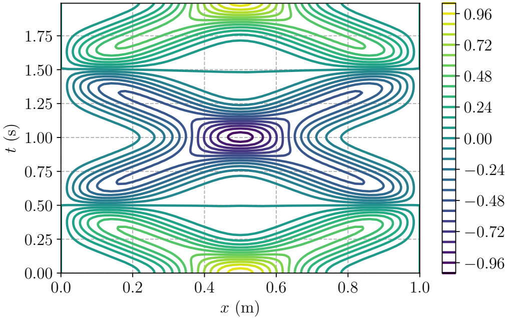

Figure 26: Response over time as a contour plot.

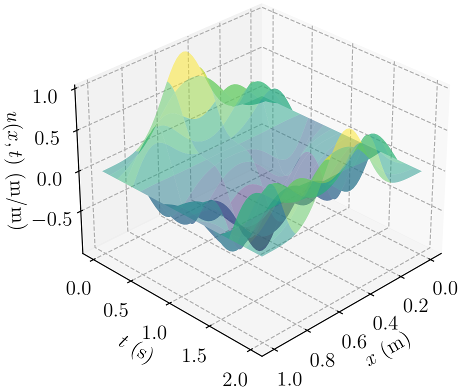

Figure 27: Response over time at a 3D plot.

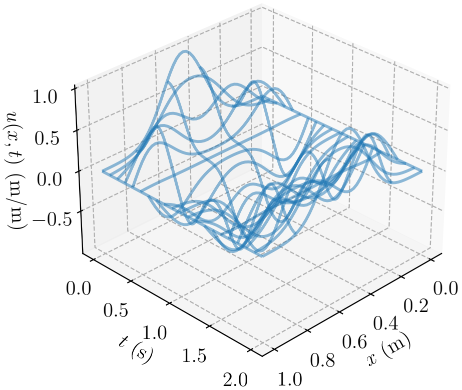

Figure 28: Wireframe response over time at a 3D plot.

$n = 100$

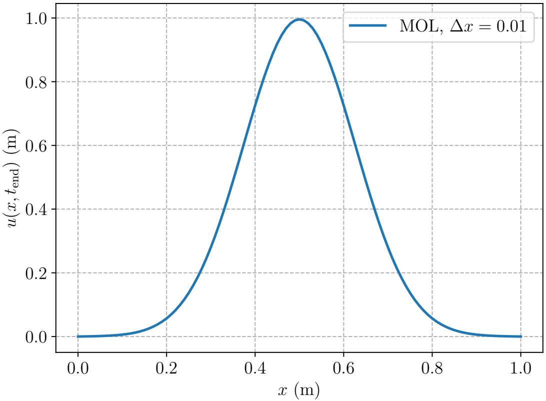

Figure 29: Response at t_{end}.

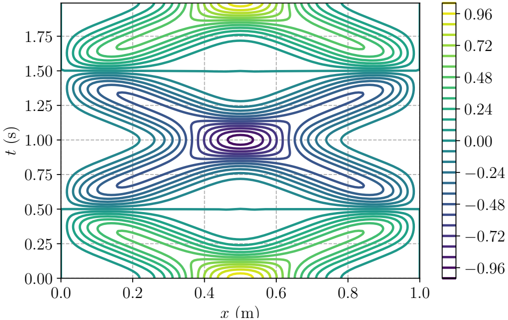

Figure 30: Response over time as a contour plot.

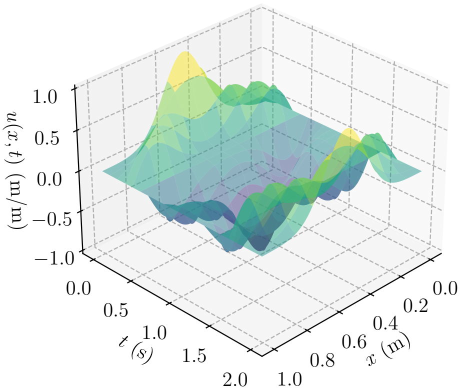

Figure 31: Response over time at a 3D plot.

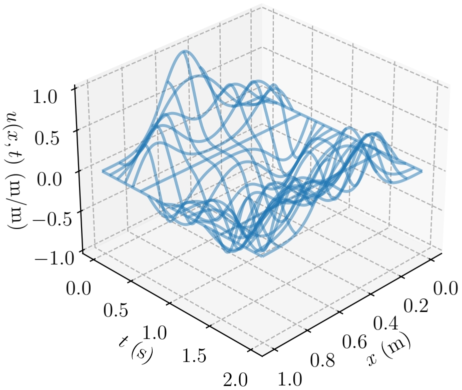

Figure 32: Wireframe response over time at a 3D plot.

Fully Discrete Methods Applied to the Diffusion Equation

NOT TAUGHT FALL 2024

- Consider the diffusion (heat) equation

$$u_t = \alpha^2 u_{xx} + f, \quad 0 < x < \ell, \quad t \geq 0, \tag{1}$$

  where $f = f(x,t)$.
- The ICs are

$$u(x,0) = \phi(x), \quad 0 < x < \ell.$$

- The BCs are

$$u(0,t) = \mu_0, \quad u(\ell,t) = \mu_\ell, \quad t \geq 0.$$

- A fully discrete method discretizes (1) in space and in time.

Discretize in Space and Time

- Discretize in space, meaning $x_\imath = \imath\Delta x$, $\Delta x = \frac{\ell}{n+1}$, $x_{\imath+1} = x_\imath + \Delta x$, $x_{\imath-1} = x_\imath - \Delta x$, $\imath = 0, 1, \ldots, n+1$.
- Discretize in time, meaning $t_k = k\Delta t$, $k = 0, 1, 2\ldots$ and Δt is sufficiently small.
- Let $q_\imath^k = u(x_\imath, t_k)$.
- Using
  - a forward-difference first derivative approximation in time to approximate u_t, and
  - a central-difference second derivative approximation to approximate in space to approximate u_{xx},

  (1) becomes

$$
\begin{aligned}
\frac{u(x_\imath, t_{k+1}) - u(x_\imath, t_k)}{\Delta t} &= \frac{\alpha^2}{(\Delta x)^2}\left(u(x_{\imath+1}, t) - 2u(x_\imath, t) + u(x_{\imath-1}, t)\right) + f(x_\imath, t_k), \quad \imath = 1, \ldots, n, \\
\frac{q_\imath^{k+1} - q_\imath^k}{\Delta t} &= \kappa\left(q_{\imath+1}^k - 2q_\imath^k + q_{\imath-1}^k\right) + f_\imath^k, \quad \imath = 1, \ldots, n, \qquad (12)
\end{aligned}
$$

  where $\kappa = \frac{\alpha^2}{(\Delta x)^2}$ and $f_\imath^k = f(x_\imath, t_k)$.
- From the ICs, $q_\imath^0 = u(x_\imath, 0) = \phi(x_\imath)$, $i = 1, \ldots, n-1, n$.
- From the BCs, $q_0^k = u(0, t_k) = \mu_0$ and $q_{n+1}^k = u(x_{n+1}, t_k) = u(\ell, t) = \mu_\ell$.

Fully Discrete Methods Applied to the Wave Equation

- Consider the wave equation

$$u_{tt} = \alpha^2 u_{xx} + f, \quad 0 < x < \ell, \quad t \geq 0, \tag{6}$$

  where $f = f(x,t)$.
- The ICs are

$$\begin{aligned} u(x,0) &= \phi(x), \quad 0 < x < \ell, \\ \dot{u}(x,0) &= \psi(x), \quad 0 < x < \ell. \end{aligned}$$

- The BCs are

$$u(0,t) = \mu_0, \quad u(\ell,t) = \mu_\ell, \quad t \geq 0.$$

- A fully discrete method discretizes (6) in space and in time.

Discretize in Space and Time

- Discretize in space, meaning $x_\imath = \imath \Delta x$, $\Delta x = \frac{\ell}{n+1}$, $x_{\imath+1} = x_\imath + \Delta x$, $x_{\imath-1} = x_\imath - \Delta x$, $\imath = 0, 1, \ldots, n+1$.
- Discretize in time, meaning $t_k = k\Delta t$, $k = 0, 1, 2\ldots$ and Δt is sufficiently small.
- Let $q_\imath^k = u(x_\imath, t_k)$.
- Using
  - a central-difference second derivative approximation in time to approximate u_{tt}, and
  - a central-difference second derivative approximation to approximate in space to approximate u_{xx},

  (6) becomes

$$\frac{u(x_\imath, t_{k+1}) - 2u(x_\imath, t_k) + u(x_\imath, t_{k-1})}{(\Delta t)^2} = \frac{\alpha^2}{(\Delta x)^2} \left(u(x_{\imath+1}, t_k) - 2u(x_\imath, t_k) + u(x_{\imath-1}, t_k)\right) + f(x_\imath, t_k), \quad \imath = 1, .$$

$$\frac{q_\imath^{k+1} - 2q_\imath^k + q_\imath^{k-1}}{(\Delta t)^2} = \kappa \left(q_{\imath+1}(t) - 2q_\imath(t) + q_{\imath-1}(t)\right) + f_\imath^k, \quad \imath = 1, \ldots, n, \tag{13}$$

  where $\kappa = \frac{\alpha^2}{(\Delta x)^2}$ and $f_\imath^k = f(x_\imath, t_k)$

- From the ICs, $q_\imath^0 = u(x_\imath, 0) = \phi(x_\imath)$, $\dot{u}(x_\imath, 0) = \psi(x_\imath) = \frac{q_\imath^1 - q_\imath^0}{\Delta t}$, $i = 1, \ldots, n-1, n$.
- From the BCs, $q_0^k = u(0, t_k) = \mu_0$ and $q_{n+1}^k = u(x_{n+1}, t_k) = u(\ell, t) = \mu_\ell$.

References

Slides based on [1, Ch. 18, 19, 20], [2, Ch. 20], [3, Ch. 11], [4, Ch. 16], [5, Ch. 12, 15].

[1] M. D. Greenberg, *Advanced Engineering Mathematics*, 2nd. Upper Saddle River, NJ: Prentice Hall, 1998.
[2] M. Yano, *AER 336: Scientific Computing Lecture Notes*. Toronto ON: University of Toronto, 2017.
[3] M. T. Heath, *Scientific Computing — An Introductory Survey*, Second. New York, NY: McGraw-Hill, 2002.
[4] J. Solomon, *Numerical Algorithms: Methods for Computer Vision, Machine Learning and Graphics*. Boca Raton, FL: CRC Press, 2015.
[5] D. G. Zill and M. R. Cullen, *Differential Equations with Boundary-Value Problems*, 6th. Toronto, ON: Thomson Brooks/Cole, 2005.