MTH 9821 Numerical Methods for Finance I
Lecture 8 –Finite Difference Method

1 Finite Difference Solution for ODEs

Find $y : [a, b] \to \mathbb{R}$ such that

$$\begin{aligned} y'(x) &= f\big(x, y(x)\big), \quad \forall x \in [a, b] \\ y(a) &= c \end{aligned}$$

Remark: Every ODE satisfies the above form.

Discretize the interval $[a, b]$ using N equal intervals
Nodes $x_i = a + ih, \quad i = 0 : N$, where $h = \frac{b-a}{N}$.
Finite difference solution requires finding approximate values for the exact solution at every nodes $x_i,\ \forall i = 0 : N$

$$\begin{aligned} \text{Find } y_1 &\text{ approximation of } y_{exact}(x_1) \\ \vdots \quad & \qquad\qquad\qquad \vdots \\ y_N &\text{ approximation of } y_{exact}(x_N) \end{aligned}$$

1.1 1st Order ODE

$$\begin{aligned} y'(x) &= f\big(x, y(x)\big), \quad \forall x \in [a, b] \\ \Rightarrow y'(x_i) &= f\big(x_i, y(x_i)\big), \quad \forall i = 1 : N \end{aligned}$$

Using forward finite difference approximation:

$$\begin{aligned} & y'(x_i) = \frac{y(x_i + h) - y(x_i)}{h} + O(h) \\ \Rightarrow\quad & \frac{y(x_i + h) - y(x_i)}{h} + O(h) = f\big(x_i, y(x_i)\big), \quad \forall i = 1 : N,\ x_i + h = x_{i+1} \\ \Rightarrow\quad & \frac{y(x_{i+1}) - y(x_i)}{h} + O(h) = f\big(x_i, y(x_i)\big) \\ \Rightarrow\quad & \frac{y_{i+1} - y_i}{h} = f\big(x_i, y_i\big), \quad \forall i = 0 : N - 1 \end{aligned}$$

(1) **Solve**

$$\begin{cases} y_{i+1} = y_i + hf(x_i, y_i), \quad \forall i = 0 : (N-1) \\ y_0 = C \end{cases}$$

Find $y_1,\ y_2, \dots, y_N$ (one step forward each)

(2) **Convergence**

Approximation error of the finite difference solution

$$e_i = y(x_i) - y_i$$

Convergent if

$$\lim_{n \to \infty} \left(\max_{i=1:N} |e_i| \right) = 0$$

Note:
After finding the discrete nodes values, we can use cubic spline interpolation to find the entire curve of y.

1.2 2nd Order ODE

$$\begin{aligned} x^2 y'' + xy' - y &= 0 \\ y'' &= -\frac{1}{x} y' + \frac{1}{x^2} y \end{aligned}$$

Question: How would a 2nd order ODE satisfy the form $y'(x) = f\big(x, y(x)\big)$?

$$\begin{aligned} \text{Define} \quad Y &= \begin{pmatrix} y \\ y' \end{pmatrix}, \quad f : \mathbb{R}^2 \to \mathbb{R}^2 \\ \text{thus} \quad Y' &= \begin{pmatrix} y' \\ y'' \end{pmatrix} = \begin{pmatrix} y' \\ -\frac{1}{x} y' + \frac{1}{x^2} y \end{pmatrix} = f(Y) = \begin{pmatrix} 0 & 1 \\ \frac{1}{x^2} & -\frac{1}{x} \end{pmatrix} \begin{pmatrix} y \\ y' \end{pmatrix} \end{aligned}$$

Exact Solution
For certain 2nd order ODEs, we can solve for the exact solution:

$$\begin{aligned} e.g., \quad & x^2 y''(x) + xy'(x) - y(x) = 0, \quad \forall 1 < x < 2 \\ & y(1) = 0 \\ & y(2) = 1.5 \end{aligned}$$

The exact solution is of the form

$$\begin{aligned} & y(x) = c_1 x^{\alpha_1} + c_2 x^{\alpha_2} \\ \text{Guess: } & y(x) = x^a \\ \text{Characteristic equation: } & a(a-1) + a - 1 = 0 \\ \Rightarrow & a = 1 \text{ or } a = -1 \\ & y(x) = c_1 x + c_2 \frac{1}{x}, \quad \text{find } c_1 \text{ and } c_2 \text{ from boundary conditions} \end{aligned}$$

Finite Difference Solution

$$\begin{aligned} e.g., \quad & (x^2+1)y''(x) + x^2y'(x) - 3xy(x) = 0, \quad \forall -1 < x < 2 \\ & y(-1) = 1 \\ & y(2) = 3 \end{aligned}$$

Discretize $[-1, 2]$ into n intervals using nodes $x_i = -1 + ih$, $\forall i = 0 : N$; $h = \frac{3}{N}$

$$(x_i^2+1)y''(x_i) + x_i^2y'(x_i) - 3x_iy(x_i) = 0, \quad \forall i = 0 : (N-1)$$

Using Central Finite Differences:

$$\begin{aligned} y''(x_i) &= \frac{y(x_i-h) - 2y(x_i) + y(x_i+h)}{h^2} + O(h^2) \\ y'(x_i) &= \frac{y(x_i+h) - y(x_i-h)}{2h} + O(h^2) \end{aligned}$$

Remark:
All finite difference approximations of different order derivatives should use the same order of convergence.
(Because the overall approximation convergence will be of the lowest level of convergence.)

$$\Rightarrow (x_i^2+1)\left(\frac{y(x_i-h) - 2y(x_i) + y(x_i+h)}{h^2} + O(h^2)\right) + x_i^2\left(\frac{y(x_i+h) - y(x_i-h)}{2h} + O(h^2)\right) - 3x_iy(x_i) = 0$$

$$\Rightarrow (x_i^2+1)(y_{i-1} - 2y_i + y_{i+1}) + hx_i^2\left(\frac{y_{i+1} - y_{i-1}}{2}\right) - 3h^2x_iy_i = 0, \quad \forall i = 1 : (N-1)$$

$$y_0 = y(-1) = 1, \quad y_N = y(2) = 3$$

$$\Rightarrow \left(x_i^2 + 1 - \frac{hx_i^2}{2}\right)y_{i-1} + \left(-2(x_i^2+1) - 3h^2x_i\right)y_i + \left(x_i^2 + 1 + \frac{hx_i^2}{2}\right)y_{i+1} = 0, \quad \forall i = 1 : (N-1)$$

Matrix form:

$$A\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N-1} \end{pmatrix} = \begin{pmatrix} -\left(x_1^2 + 1 - \frac{hx_1^2}{2}\right)y_0 \\ 0 \\ \vdots \\ 0 \\ -\left(x_{N-1}^2 + 1 - \frac{hx_{N-1}^2}{2}\right)y_N \end{pmatrix}$$

$$\begin{aligned} y_0 = 1, \quad x_1 = -1 + h \\ y_N = 3, \quad x_{N-1} = 2 - h \end{aligned}$$

where

$$A = \begin{pmatrix} -2(x_1^2+1) - 3h^2x_1 & x_1^2 + 1 + \frac{hx_1^2}{2} & 0 & 0 & \cdots & 0 \\ x_2^2 + 1 - \frac{hx_2^2}{2} & -2(x_2^2+1) - 3h^2x_2 & x_2^2 + 1 + \frac{hx_2^2}{2} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & x_{N-1}^2 + 1 - \frac{hx_{N-1}^2}{2} & -2(x_{N-1}^2+1) - 3h^2x_{N-1} \end{pmatrix}$$

Remark:
- ⋆ If boundary condition is given as $y'(-1) = -2$, then we need new linear condition.
- ⋆ If we double the number of nodes, the approximation error should be reduced by 4 times, no matter how complex the ODE might be.
  In practice, the number should be really close to 4. If it turns out to be somewhere around 3.8, there might very well be a bug in your code.

2 Finite Difference Solutions of the Heat PDE

Heat PDE:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}, \quad t > 0,\ x \in \mathbb{R}$$

Boundary condition:

$$u(x,0) = f(x), \quad \forall x \in \mathbb{R}$$

Heat PDE on a bounded domain:

$$u_t = u_{xx} \quad \forall t > 0,\ \forall x_{left} < x < x_{right}$$

Boundary condition:

$$\begin{aligned} &u(x,0) = f(x), \quad \forall x_{left} < x < x_{right} \\ &u(x_{left},\tau) = g_{left}(\tau), \quad \forall \tau > 0 \\ &u(x_{right},\tau) = g_{right}(\tau), \quad \forall \tau > 0 \end{aligned}$$

Find a finite difference approximation of $u(x, \tau_{final})$

2.1 Domain Discretization

- Discretize in x direction by using N intervals of size

  $$\delta_x = \frac{x_{right} - x_{left}}{N}$$

  corresponding to nodes $x_n = x_{left} + n\delta_x$, $\forall n = 0 : N$

- Discretize in τ direction by using M intervals of size

  $$\delta_\tau = \frac{\tau_{final}}{M}$$

  corresponding to nodes $\tau_m = m\delta_t$, $\forall m = 0 : M$

Goal:
For nodes (x_n, τ_m), find approximate values of $u(x_n, \tau_m)$, denoted u_n^m.
Finite difference solution requires finding u_n^M for $n = 0 : N$ Fix m, write a discretization of

$$\frac{\partial u}{\partial \tau}(x_n, \tau_m) = \frac{\partial^2 u}{\partial x^2}(x_n, \tau_m), \quad \forall n = 1 : (N-1)$$

Three finite difference discretization methods:

- Forward Euler — explicit method
- Backward Euler — implicit method
- Crank-Nicolson — implicit method

2.2 Forward Euler

Use forward finite difference approximation for u_τ and central finite difference approximation for u_{xx}:
Fix m,

$$\begin{aligned}
&\frac{\partial u}{\partial \tau}(x_n, \tau_m) = \frac{u(x_n, \tau_{m+1}) - u(x_n, \tau_m)}{\delta_\tau} + O(\delta_\tau)\\
&\frac{\partial^2 u}{\partial x^2}(x_n, \tau_m) = \frac{u(x_{n+1}, \tau_m) - 2u(x_n, \tau_m + u(x_{n-1}, \tau_m))}{(\delta_x)^2} + O\big((\delta_x)^2\big)\\
\Rightarrow\quad &\frac{u_n^{m+1} - u_n^m}{\delta_\tau} = \frac{u_{n-1}^m - 2u_n^m + u_{n+1}^m}{(\delta_x)^2}\\
\Rightarrow\quad &u_n^{m+1} - u_n^m = \alpha\big(u_{n-1}^m - 2u_n^m + u_{n+1}^m\big)\\
\Rightarrow\quad &u_n^{m+1} = \alpha u_{n-1}^m + (1-2\alpha)u_n^m + \alpha u_{n+1}^m, \quad \forall n = 1:(N-1) \qquad (\star)
\end{aligned}$$

Given u_n^m for all $n = 0:N$, use $(\star)$ to find u_n^{m+1} for all $n = 1:(N-1)$

Remark:
$\star$ The essence of Forward Euler: one step forward in time for each given m.

Note:

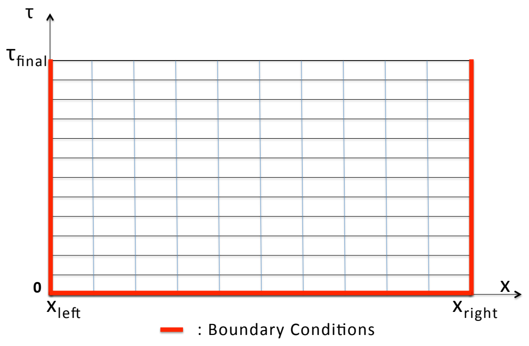
- In order to cancel out the big O terms, we want to keep

$$\frac{\delta_\tau}{(\delta_x)^2} = \alpha$$

  as constant, which is also called the "Courant constant".

- If we double the number of intervals in x, i.e., $N \to 2N$, $\delta_x \to \frac{\delta_x}{2}$. We need $\delta_\tau \to \frac{\delta_\tau}{4}$, i.e., $M \to 4M$.

Boundary Conditions:



Note: we have boundary conditions on red regions, i.e.,

$$\begin{aligned} u_0^m &= u(x_0,\tau_m) = u(x_{left}, m\delta_\tau) = g_{left}(m\delta_\tau) \\ u_N^m &= u(x_N,\tau_m) = u(x_{right}, m\delta_\tau) = g_{right}(m\delta_\tau) \\ u_n^0 &= u(x_n,0) = f(x_n) = f(x_{left}+n\delta_x) \\ \text{We need}\quad f(x_{left}) &= g_{left}(0) \\ f(x_{right}) &= g_{right}(0) \end{aligned}$$

Remark:
⋆ Forward Euler is an **explicit method**, since we don't need to solve linear system.

$$\text{Let}\quad U^m = \underbrace{\begin{pmatrix} u_1^m \\ \vdots \\ u_{N-1}^m \end{pmatrix}}_{(N-1)\times 1}, \quad \text{we have}\quad U^{m+1} = AU^m + \underbrace{\begin{pmatrix} \alpha u_0^m \\ 0 \\ \vdots \\ 0 \\ \alpha u_N^m \end{pmatrix}}_{(N-1)\times 1}$$

$$\text{where } A = \underbrace{\begin{pmatrix} 1-2\alpha & \alpha & 0 & 0 & \dots & 0 \\ \alpha & 1-2\alpha & \alpha & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \alpha & 1-2\alpha & \alpha \\ 0 & \dots & \dots & 0 & \alpha & 1-2\alpha \end{pmatrix}}_{(N-1)\times(N-1)}$$

where $u_0^m = g_{left}(m\delta_\tau)$, $u_N^m = g_{right}(m\delta_\tau)$.

Pseudocode: *Forward Euler for Heat PDE*

```
for $n = 1 : (N-1)$
    $U(n) = f(x_{left} + n\delta_x)$;
    $A(n,n) = 1 - 2\alpha$;
end
for $n = 2 : (N-1)$
    $A(n, n-1) = \alpha$;
    $A(n-1, n) = \alpha$;
end
for $m = 0 : (M-1)$
    $b(1) = g_{left}(m\delta_\tau)$;
    $b(N-1) = g_{right}(m\delta_\tau)$;
    $U = AU + b$;
end
```

Remark:
⋆ Explicit method: no linear solving required.
⋆ Convergence condition: $\alpha \leq \frac{1}{2}$.

2.3 Backward Euler

$$\frac{\partial u}{\partial \tau}(x_n, \tau_{m+1}) = \frac{\partial^2 u}{\partial x^2}(x_n, \tau_{m+1}), \quad \forall n = 1 : (N-1)$$

Use backward finite difference approximation for u_τ and central finite difference approximation for u_{xx}:
Fix m,

$$\begin{aligned}
&\frac{\partial u}{\partial \tau}(x_n, \tau_{m+1}) = \frac{u(x_n, \tau_{m+1}) - u(x_n, \tau_m)}{\delta_\tau} + O(\delta_\tau) \\
&\frac{\partial^2 u}{\partial x^2}(x_n, \tau_{m+1}) = \frac{u(x_{n+1}, \tau_{m+1}) - 2u(x_n, \tau_{m+1} + u(x_{n-1}, \tau_{m+1}))}{(\delta_x)^2} + O\big((\delta_x)^2\big) \\
\Rightarrow \quad &\frac{u_n^{m+1} - u_n^m}{\delta_\tau} = \frac{u_{n-1}^{m+1} - 2u_n^{m+1} + u_{n+1}^{m+1}}{(\delta_x)^2} \\
&\text{Keep } \frac{\delta_\tau}{(\delta_x)^2} = \alpha \quad \text{as constant.} \\
\Rightarrow \quad &u_n^{m+1} - u_n^m = \alpha\big(u_{n-1}^{m+1} - 2u_n^{m+1} + u_{n+1}^{m+1}\big) \\
\Rightarrow \quad &-\alpha u_{n-1}^{m+1} + (1+2\alpha)u_n^{m+1} - \alpha u_{n+1}^{m+1} = u_n^m, \quad \forall n = 1 : (N-1) \qquad (\star)
\end{aligned}$$

Given u_n^m for all $n = 0 : N$, use $(\star)$ to find u_n^{m+1} for all $n = 1 : (N-1)$.

$$\text{Let} \quad U^m = \underbrace{\begin{pmatrix} u_1^m \\ \vdots \\ u_{N-1}^m \end{pmatrix}}_{(N-1)\times 1}, \quad \text{we have} \quad AU^{m+1} = U^m + \underbrace{\begin{pmatrix} \alpha u_0^{m+1} \\ 0 \\ \vdots \\ 0 \\ \alpha u_N^{m+1} \end{pmatrix}}_{(N-1)\times 1}$$

$$\text{where } A = \underbrace{\begin{pmatrix} 1+2\alpha & -\alpha & 0 & 0 & \dots & 0 \\ -\alpha & 1+2\alpha & -\alpha & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & -\alpha & 1+2\alpha & -\alpha \\ 0 & \dots & \dots & 0 & -\alpha & 1+2\alpha \end{pmatrix}}_{(N-1)\times(N-1)}$$

where $u_0^{m+1} = g_{left}\big((m+1)\delta_\tau\big)$, $u_N^{m+1} = g_{right}\big((m+1)\delta_\tau\big)$.

Pseudocode: *Backward Euler for Heat PDE*

```
$[LL, UU] = lu(A);$
      (Note: A is a spd tridiagonal matrix, LU decomposition is more efficient than Cholesky)
Compute $U^0$ from boundary conditions;
for $m = 0 : (M-1)$
    $y = forward\_subst\Big(LL, U^m + \begin{pmatrix} \alpha u_0^{m+1} \\ 0 \\ \vdots \\ 0 \\ \alpha u_N^{m+1} \end{pmatrix}\Big);$
    $U^{m+1} = backward\_subst(UU, y);$
end
```

Remark:
⋆ Implicit method.
⋆ Convergent for any α, but is slower than Forward Euler.

2.4 Crank-Nicolson

$$\frac{\partial u}{\partial \tau}(x_n, \tau_{m+\frac{1}{2}}) = \frac{\partial^2 u}{\partial x^2}(x_n, \tau_{m+\frac{1}{2}}), \quad \forall n = 1:(N-1)$$

Use central finite difference approximation for u_τ:

Fix m,

$$\frac{\partial u}{\partial \tau}(x_n, \tau_{m+\frac{1}{2}}) = \frac{u(x_n, \tau_{m+1}) - u(x_n, \tau_m)}{2\frac{\delta_\tau}{2}} + O\big((\delta_\tau)^2\big)$$

$$\frac{\partial^2 u}{\partial x^2}(x_n, \tau_{m+\frac{1}{2}})$$

$$= \frac{1}{2}\left[\frac{u(x_{n+1}, \tau_{m+1}) - 2u(x_n, \tau_{m+1} + u(x_{n-1}, \tau_{m+1}))}{(\delta_x)^2} + \frac{u(x_{n+1}, \tau_m) - 2u(x_n, \tau_m + u(x_{n-1}, \tau_m))}{(\delta_x)^2}\right] + O\big((\delta_x)^4\big)$$

$$\Rightarrow \quad \frac{u_n^{m+1} - u_n^m}{\delta_\tau} = \frac{1}{2}\left[\frac{u_{n-1}^{m+1} - 2u_n^{m+1} + u_{n+1}^{m+1}}{(\delta_x)^2} + \frac{u_{n-1}^m - 2u_n^m + u_{n+1}^m}{(\delta_x)^2}\right]$$

Keep $\dfrac{\delta_\tau}{(\delta_x)^2} = \alpha$ as constant.

$$\Rightarrow \quad u_n^{m+1} - u_n^m = \frac{\alpha}{2}\left[\frac{u_{n-1}^{m+1} - 2u_n^{m+1} + u_{n+1}^{m+1}}{(\delta_x)^2} + \frac{u_{n-1}^m - 2u_n^m + u_{n+1}^m}{(\delta_x)^2}\right]$$

$$\Rightarrow \quad -\frac{\alpha}{2}u_{n-1}^{m+1} + (1+\alpha)u_n^{m+1} - \frac{\alpha}{2}u_{n+1}^{m+1} = \frac{\alpha}{2}u_{n-1}^m + (1-\alpha)u_n^m + \frac{\alpha}{2}u_{n+1}^m, \quad \forall n = 1:(N-1) \qquad (\star)$$

Given u_n^m for all $n = 0:N$, use $(\star)$ to find u_n^{m+1} for all $n = 1:(N-1)$.

$$\text{Let} \quad U^m = \underbrace{\begin{pmatrix} u_1^m \\ \vdots \\ u_{N-1}^m \end{pmatrix}}_{(N-1)\times 1}, \quad \text{we have} \quad AU^{m+1} = BU^m + \underbrace{\begin{pmatrix} \frac{\alpha}{2}u_0^{m+1} + \frac{\alpha}{2}u_0^m \\ 0 \\ \vdots \\ 0 \\ \frac{\alpha}{2}u_N^{m+1} + \frac{\alpha}{2}u_N^m \end{pmatrix}}_{(N-1)\times 1}$$

$$\text{where } A = \underbrace{\begin{pmatrix} 1+\alpha & -\frac{\alpha}{2} & 0 & 0 & \dots & 0 \\ -\frac{\alpha}{2} & 1+\alpha & -\frac{\alpha}{2} & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & -\frac{\alpha}{2} & 1+\alpha & -\frac{\alpha}{2} \\ 0 & \dots & \dots & 0 & -\frac{\alpha}{2} & 1+\alpha \end{pmatrix}}_{(N-1)\times(N-1)}, \quad B = \underbrace{\begin{pmatrix} 1-\alpha & \frac{\alpha}{2} & 0 & 0 & \dots & 0 \\ \frac{\alpha}{2} & 1-\alpha & \frac{\alpha}{2} & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \frac{\alpha}{2} & 1-\alpha & \frac{\alpha}{2} \\ 0 & \dots & \dots & 0 & \frac{\alpha}{2} & 1-\alpha \end{pmatrix}}_{(N-1)\times(N-1)}$$

where $u_0^{m+1} = g_{left}\big((m+1)\delta_\tau\big)$, $u_N^{m+1} = g_{right}\big((m+1)\delta_\tau\big)$, $u_0^m = g_{left}(m\delta_\tau)$, $u_N^m = g_{right}(m\delta_\tau)$.

Note that $A = I + \frac{\alpha}{2}T_N$ is a spd tridiagonal matrix.

```
Pseudocode:   Crank-Nicolson for Heat PDE
[LL, UU] = lu(A);
      (Note: A is a spd tridiagonal matrix, LU decomposition is more efficient than Cholesky)
Compute $U^0$ from boundary conditions;
for $m = 0 : (M-1)$
    $b^{m+1} = BU^m$;
    $b^{m+1}(1) = b^{m+1}(1) + \frac{\alpha}{2} g_{left}\big((m+1)\delta_\tau\big) + \frac{\alpha}{2} g_{left}(m\delta_\tau)$;
    $b^{m+1}(N-1) = b^{m+1}(N-1) + \frac{\alpha}{2} g_{right}\big((m+1)\delta_\tau\big) + \frac{\alpha}{2} g_{right}(m\delta_\tau)$;
    $y = forward\_subst\Big(LL, b^{m+1}\Big)$;
    $U^{m+1} = backward\_subst(UU, y)$;
end
```

Remark:
⋆ Implicit method.
⋆ Second-order convergent for any α.

Question: which method should we use?

1st Forward Euler;

2nd Crank-Nicolson (converges quadratically);

- if forward is too slow
- legacy issues

3 Pointwise Approximation and Root-Mean-Squared (RMS) Error

By discretization, we have nodes $x_k = x_{left} + k\delta_x, \quad k = 0 : N$.
The pointwise relative error of the finite difference solution is

$$error\_pointwise = \Big| u_{approx}(x_k, \tau_{final}) - u_{exact}(x_k, \tau_{final}) \Big|.$$

Recall that $u_{exact}(x_k, \tau_{final})$ is the exact value of the solution of the heat equation, $u_{exact}(x, \tau) = e^{x+\tau}$.
The maximum pointwise approximation error is defined as:

$$max\_pointwise\_error = \max_{k=1:(N-1)} \Big| u_{approx}(x_k, \tau_{final}) - u_{exact}(x_k, \tau_{final}) \Big|$$

The RMS error is defined as

$$error\_RMS = \sqrt{\frac{1}{N+1} \sum_{0 \le k \le N} \frac{\big| u_{approx}(x_k, \tau_{final}) - u_{exact}(x_k, \tau_{final}) \big|^2}{\big| u_{exact}(x_k, \tau_{final}) \big|^2}}$$