Initiating Kinetic Plasma Simulation

Prof. Dr. Mohammed Shihab

Plasma Technology

جامعة السلام
AL SALAM UNIVERSITY

Charged Particle in Electric Field

- **Equation of motion:**

$m\, d^2x/dt^2 = q\, E(x)$ & $E(x) = -k\, x$

Result: Harmonic Oscillator

- **Equivalent to Equation of motion:**

$d^2x/dt^2 + \omega^2 x = 0$ & $\omega = \mathrm{sqrt}(q\, k/m)$

- **Represents oscillatory motion**

Exact and Euler Solution

- **Exact Solution: Motion is periodic**

$$x(t) = x0 \cos(\omega t) + (v0/\omega) \sin(\omega t)$$

- **Euler: Convert to first-order system:**

$$dx/dt = v,$$
$$dv/dt = a,$$

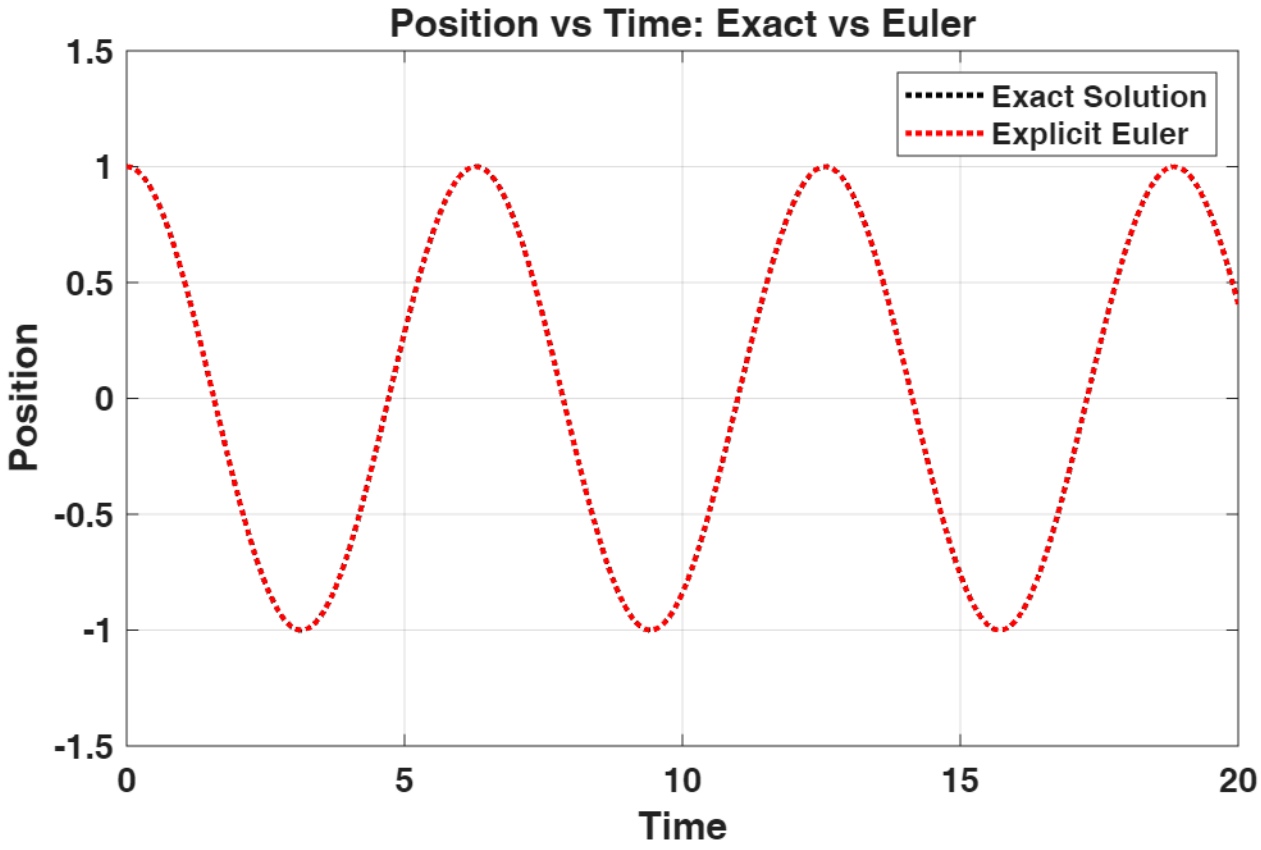
Discretization

$$v(n+1) = v(n) + a(n)\, dt$$
$$x(n+1) = x(n) + v(n+1)\, dt$$

(semi-explicit)

Where: $a(n) = -(q\, k/\, m)\, x(n)$

Exact Solution Vs Numerical Solution



- **Position of the particle: dt=0.001**

Exact: sinusoidal motion

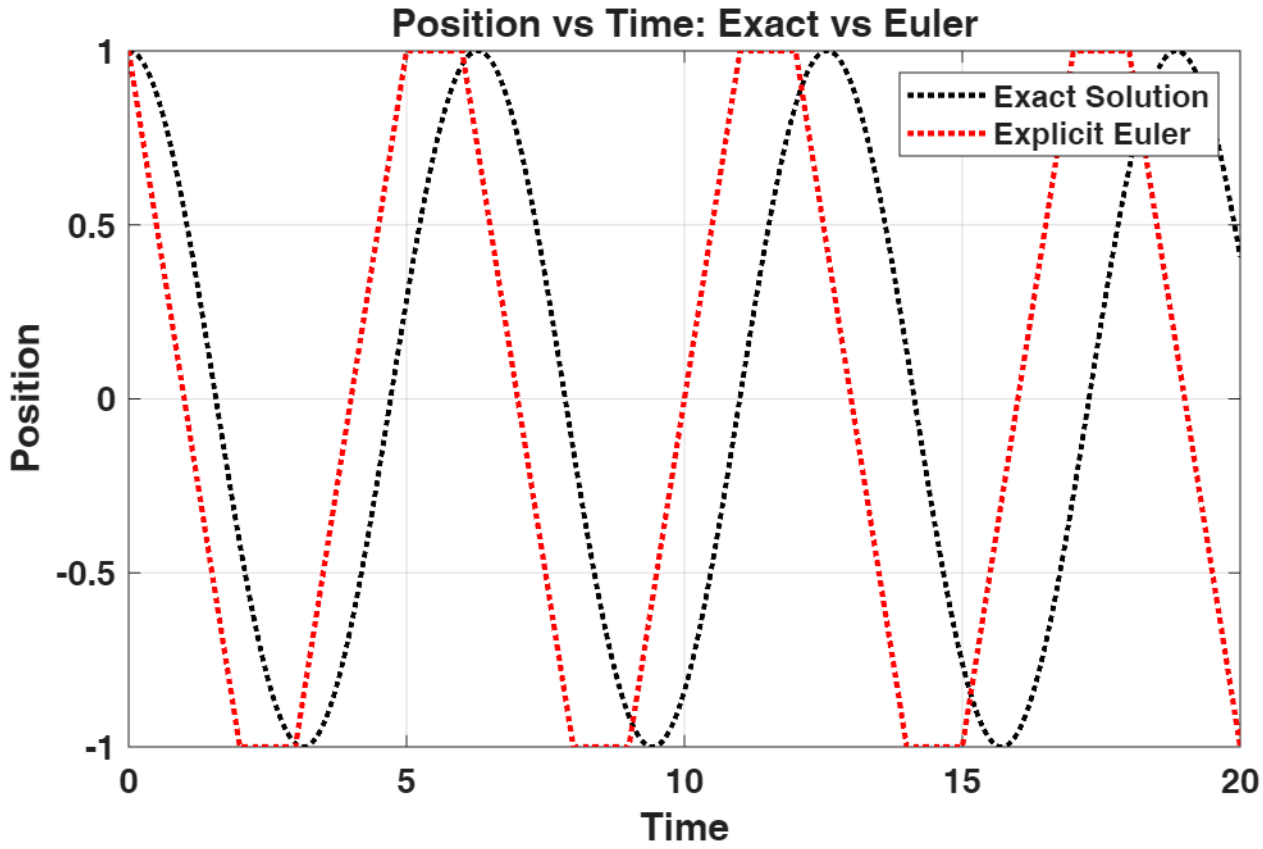
Euler: very close to the exact solution

Exact Solution Vs Numerical Solution

- **Advantage:**
  - Simple and easy
  - Low computational cost
  - Good for basic problems

- **Disadvantage:**
  - Low accuracy
  - Energy not conserved (numerical heating)
  - Unstable for oscillations

Exact Solution Vs Numerical Solution



- **Larger dt: dt=1**

Euler: phase and amplitude errors

Deviation increases with time

Numerical Instability

- **Advantage:**
  - Condition: $\omega\, dt << 1$
  - Large dt leads to instability
  - Causes artificial heating

- **Disadvantage:**
  - Low accuracy
  - Energy not conserved
  - Unstable for oscillations

Numerical Errors

❑ **Taylor Expansion**

$x(t+dt) = x(t) + dt\, x'(t) + (dt^2/2)\, x''(t) + O(dt^3)$

Remember: $v(n+1) = v(n) + a(n)\, dt$

$x(n+1) = x(n) + v(n+1)\, dt$

Euler keeps only first two terms

❑ **Local Truncation Error**

Ignored term: $(dt^2/2)\, x''(t)$

Local error per step ~ $O(dt^2)$

❑ **Global Error**

Number of steps $N = T/dt$

Global error = N × local error = $(T/dt)(dt^2) = O(dt)$

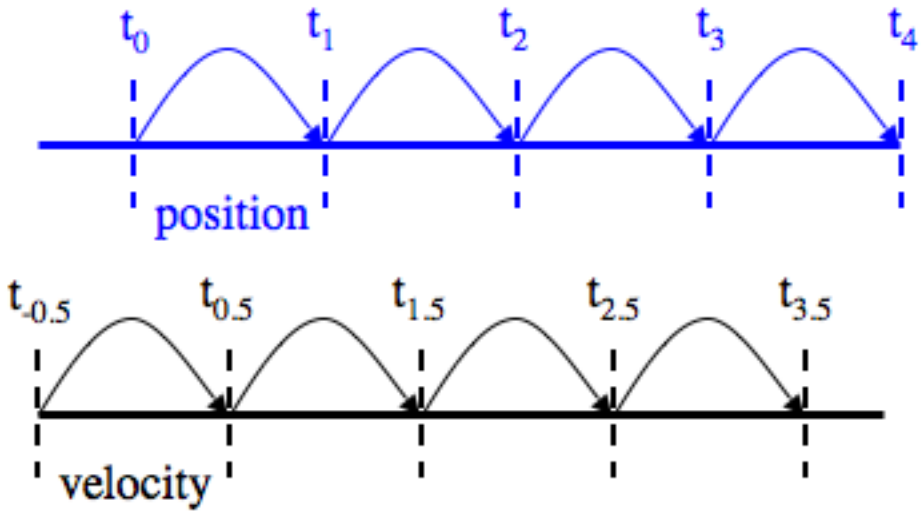
Leapfrog Method

- Explicit time integration method
- Widely used for charged particle dynamics
- Better than Euler for oscillatory systems
- Time Staggering

$$x^n = x(n\Delta t)$$

$$v^{n+1/2} = v((n+1/2)\,\Delta t)$$

Improves stability and accuracy.



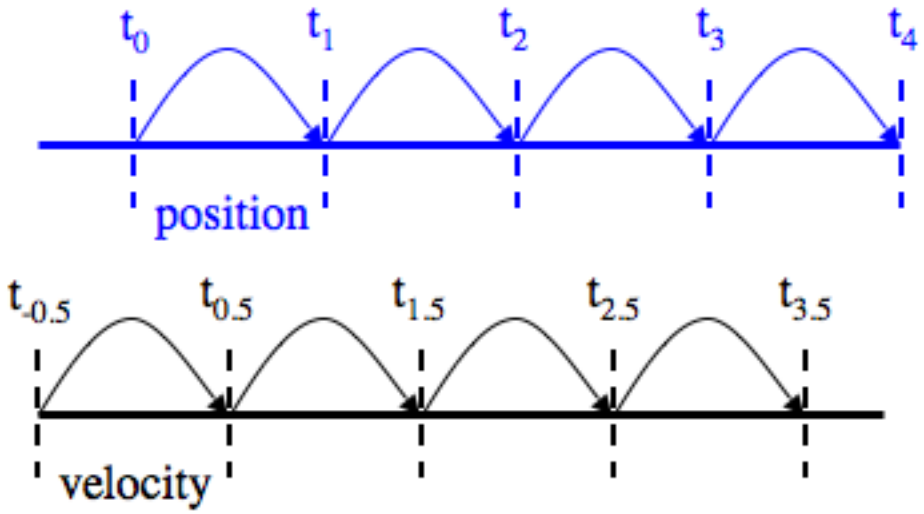
Leapfrog Method

$$V^{n+1/2} = v^{n-1/2} + a(x^n)\,\Delta t$$

$$x^{n+1} = x^n + v^{n+1/2}\Delta t$$

- Need initial half-step velocity

$$v^{1/2} = v^0 + (\Delta t/2)\, a(x^0)$$



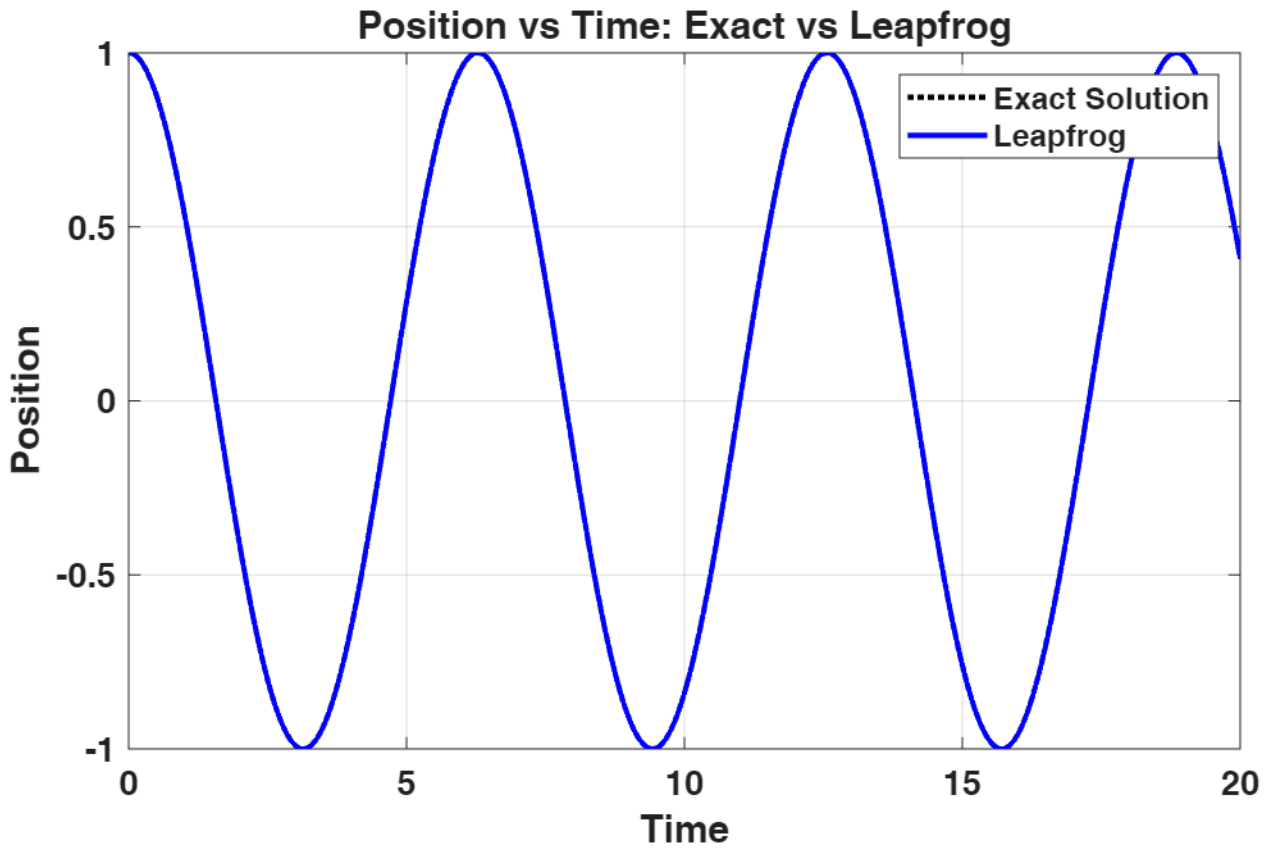
Leapfrog Method

- Second-order accuracy
- Better energy behavior
- Time-centered scheme
- Stable for oscillatory motion

  - Euler: first-order, energy drift
  - Leapfrog: second-order, stable
  - Much better for plasma simulations

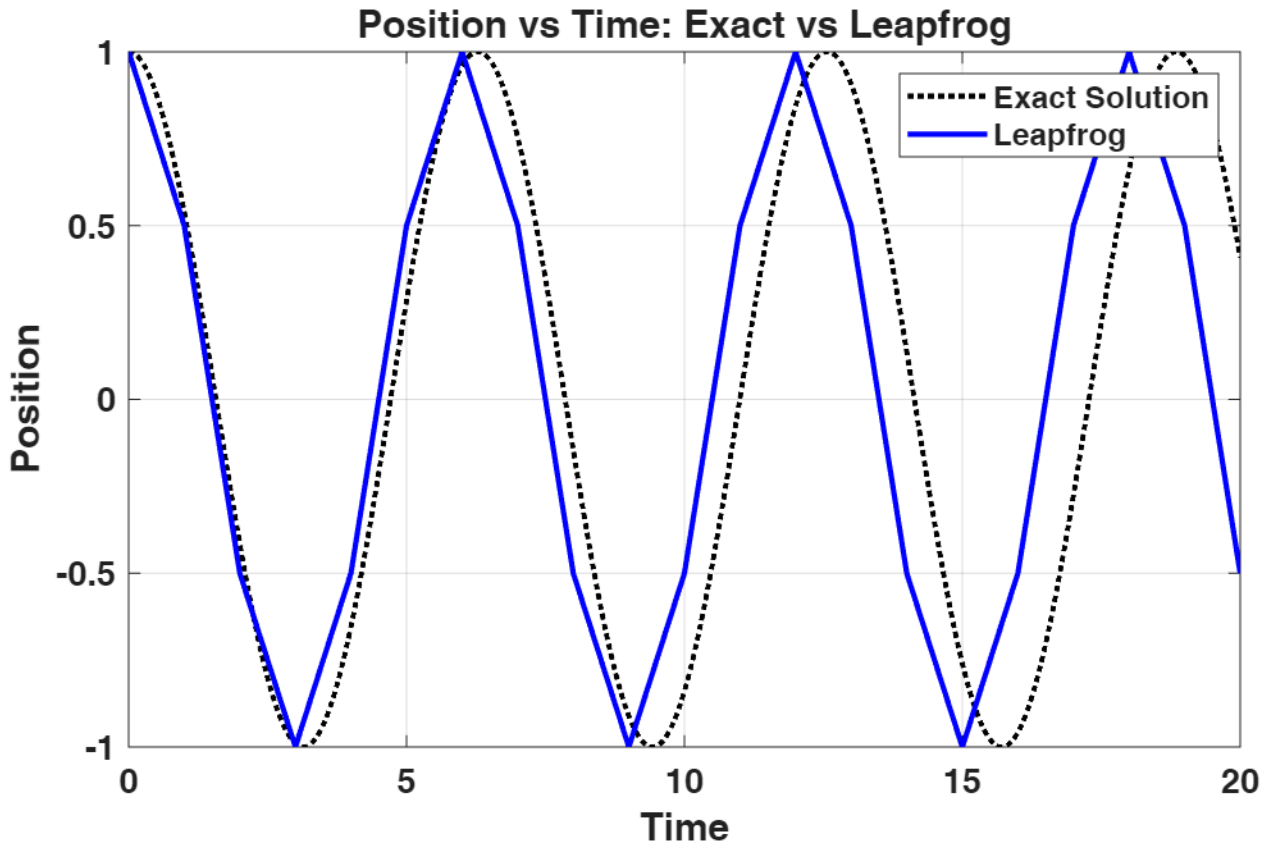
- For harmonic oscillator: $\omega\Delta t < 2$
- Too large $\Delta t \rightarrow$ instability

Leapfrog Method



- $\Delta t \rightarrow 0.001$

Leapfrog Method



- $\Delta t \rightarrow 1$

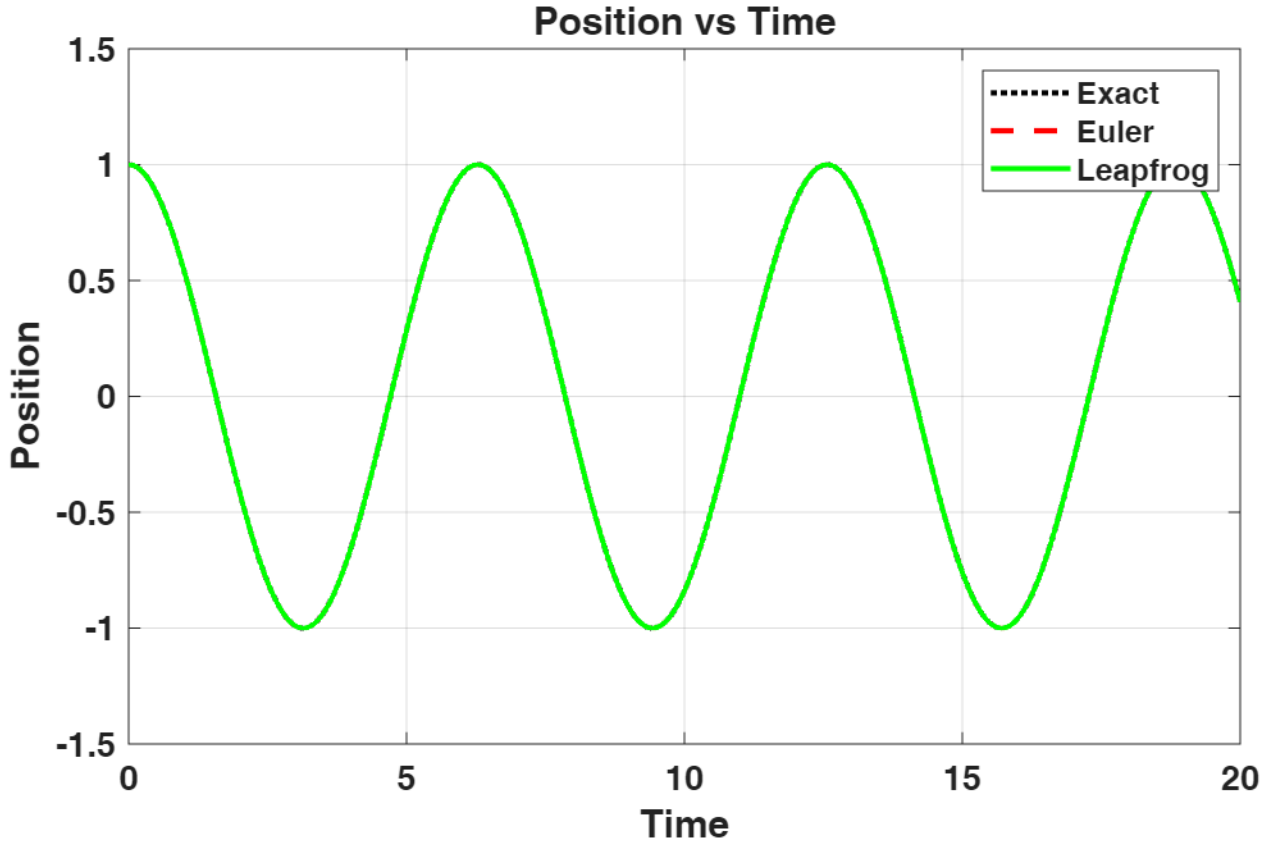
Leapfrog Method

$$x(t+\Delta t) = x + v\Delta t + (\Delta t^2/2)\, a + O(\Delta t^3)$$

- Difference appears at $O(\Delta t^3)$
  - Local error per step ~ $O(\Delta t^3)$
- Number of steps $N = T/\Delta t$
  - Global error = N × local error
    $= (T/\Delta t)(\Delta t^3) = O(\Delta t^2)$

Euler: uses slope at start → $O(\Delta t)$
Leapfrog: uses midpoint → $O(\Delta t^2)$

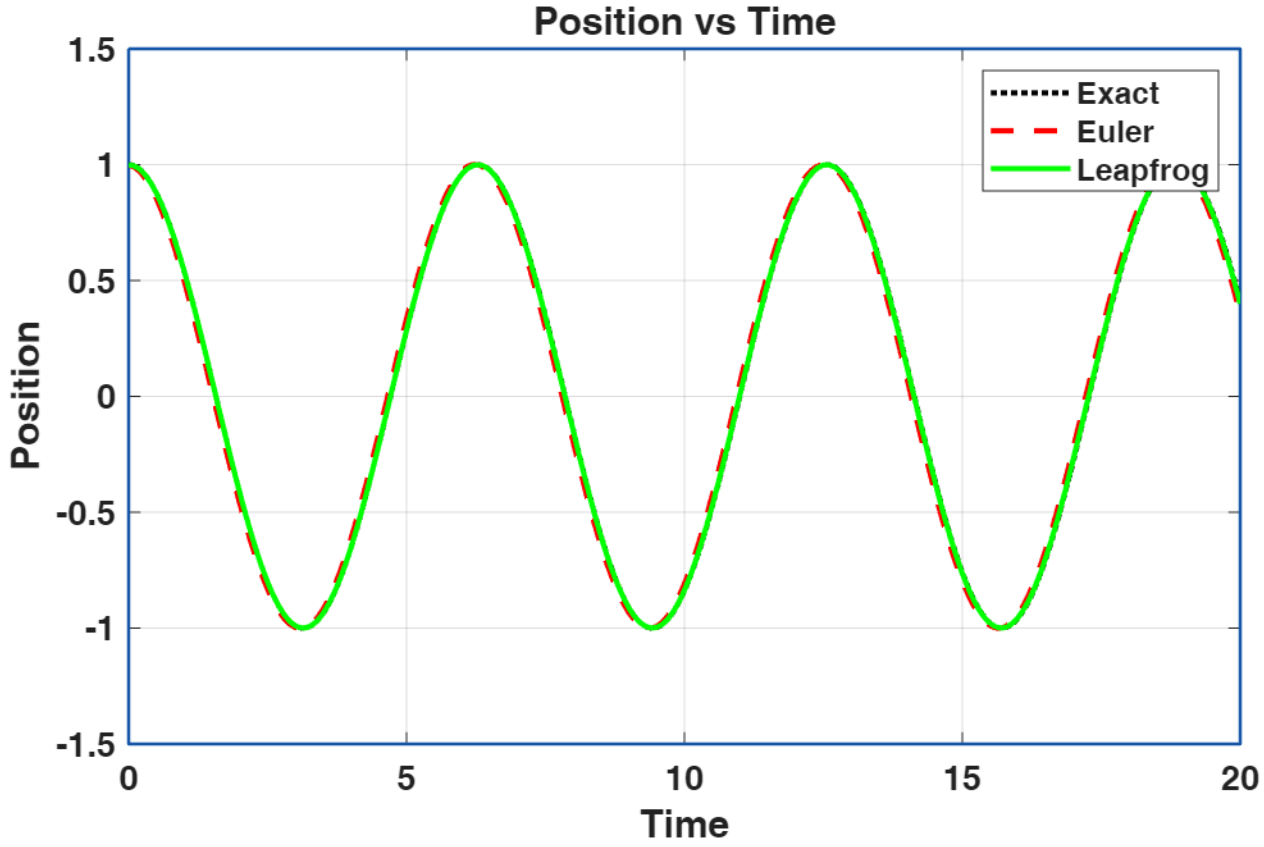
Euler vs Leapfrog



- dt=0.001

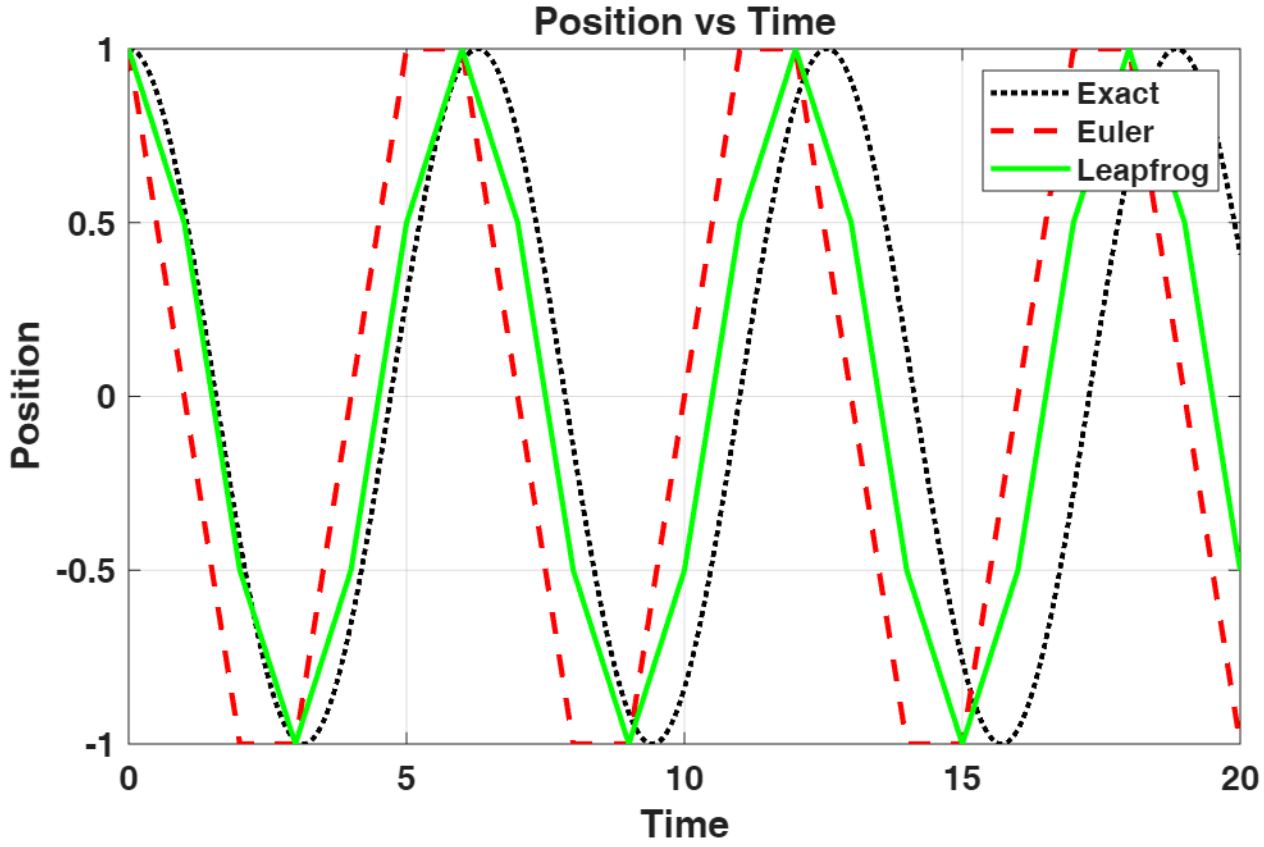
جامعة السلام
AL SALAM UNIVERSITY

Euler vs Leapfrog



- dt=0.1

Euler vs Leapfrog (explicit)



- dt=1

Explicit vs implicit Euler

- Explicit Euler (Forward):

$v(n+1) = v(n) + a(n)\,\Delta t$

$x(n+1) = x(n) + v(n)\,\Delta t$

Uses current state only (no solve)

Implicit Euler (Backward):

$v(n+1) = v(n) + a(n+1)\,\Delta t$

$x(n+1) = x(n) + v(n+1)\,\Delta t$

Requires solving for future state

Explicit vs implicit Euler

Both methods: first-order accuracy (global error $O(\Delta t)$)

Explicit Euler: Unstable for oscillations

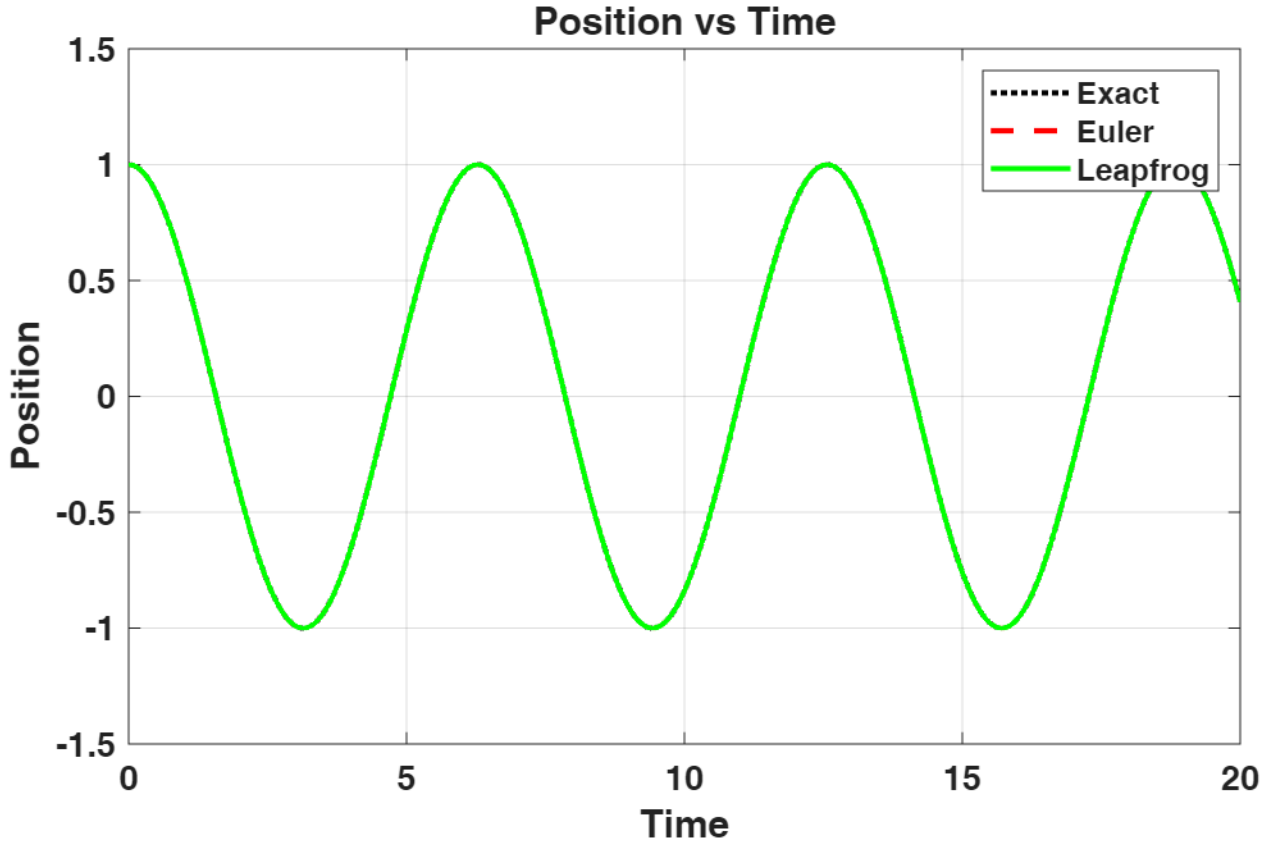
- Energy increases (numerical heating)
- Phase error grows

Implicit Euler: Unconditionally stable

- Energy decreases (numerical damping)
- Oscillations decay

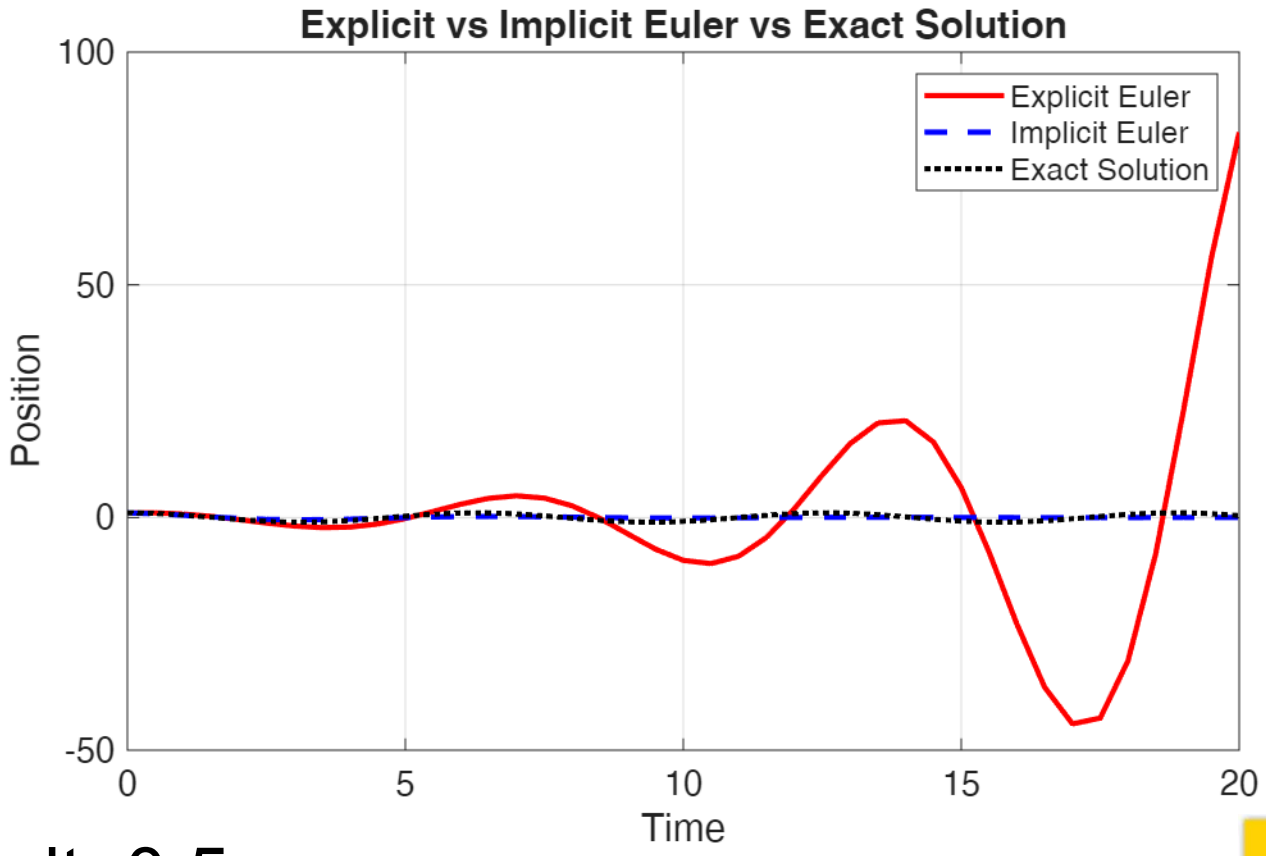
Conclusion: same order, opposite energy behavior

Implicit vs explicit Euler schemes



- dt=0.001

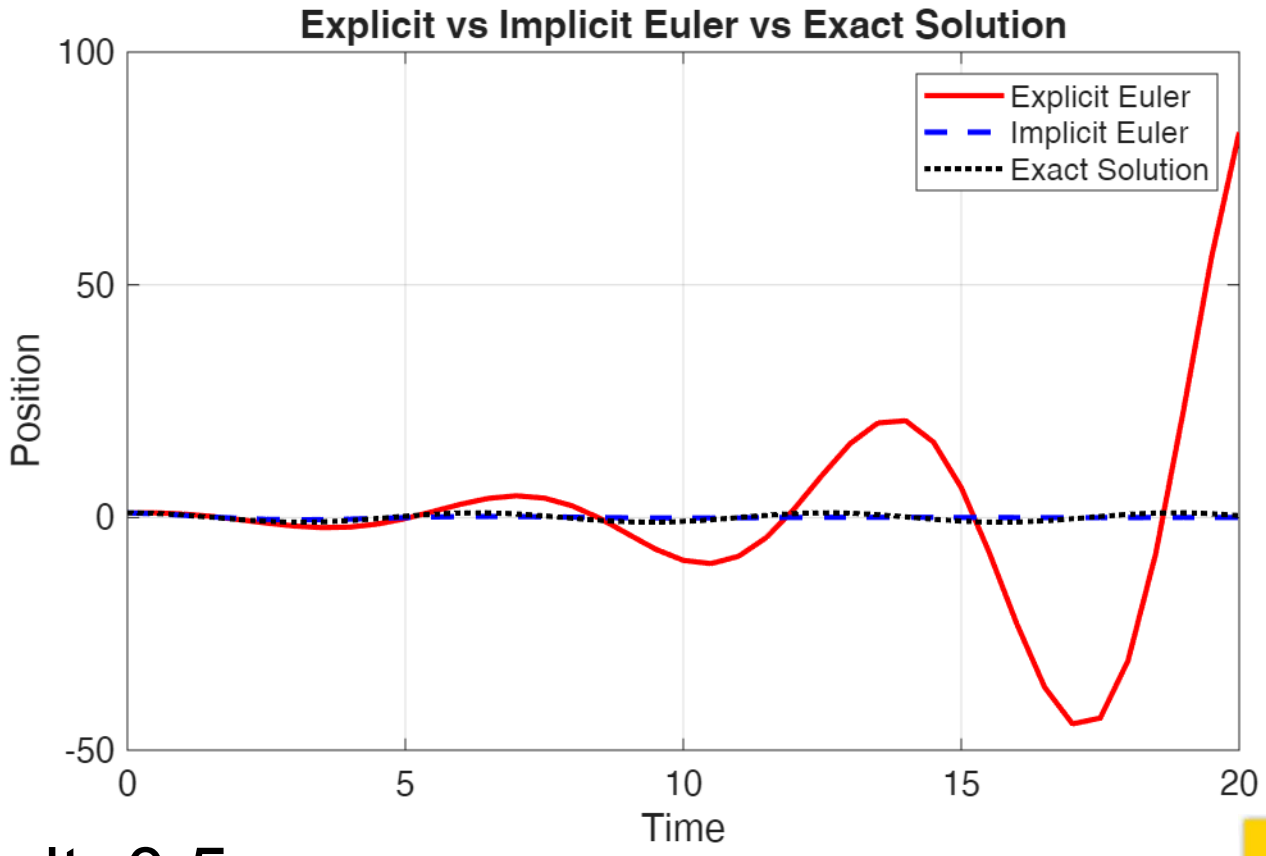
Implicit vs explicit Euler schemes



- dt=0.5
- Do not do simulation without experiments or analytical models

جامعة السلام
AL SALAM UNIVERSITY

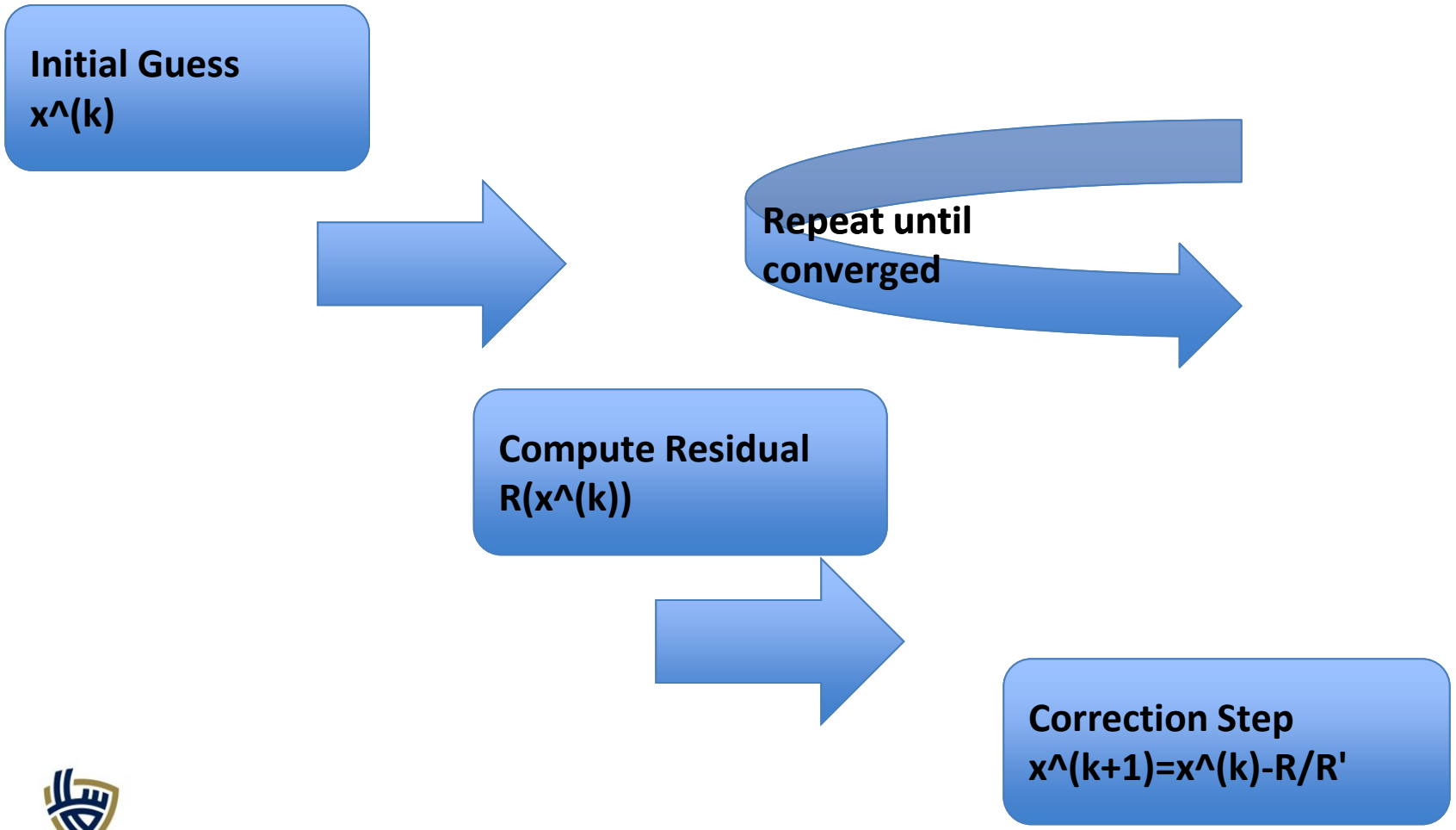
Implicit vs explicit Euler schemes



- dt=0.5
- Do not do simulation without experiments or analytical models

جامعة السلام
AL SALAM UNIVERSITY

Newton Method Flow (Implicit Scheme)



Explicit vs implicit Euler

- Implicit Euler: $y(n+1) = y(n) + \Delta t\, f(y(n+1))$

Unknown $y(n+1)$ appears inside $f(\cdot)$
Rearrange
$\rightarrow R(y(n+1)) = y(n+1) - y(n) - \Delta t\, f(y(n+1)) = 0$

- This is a nonlinear equation

We must solve for $y(n+1)$ at every time step

Newton-Raphson method

- Newton iteration: $y(k+1) = y(k) - R(y(k))/R'(y(k))$
- Efficient for nonlinear equations
- Fast (quadratic) convergence near solution
- Works for general nonlinear forces (plasma physics)

Newton-Raphson method

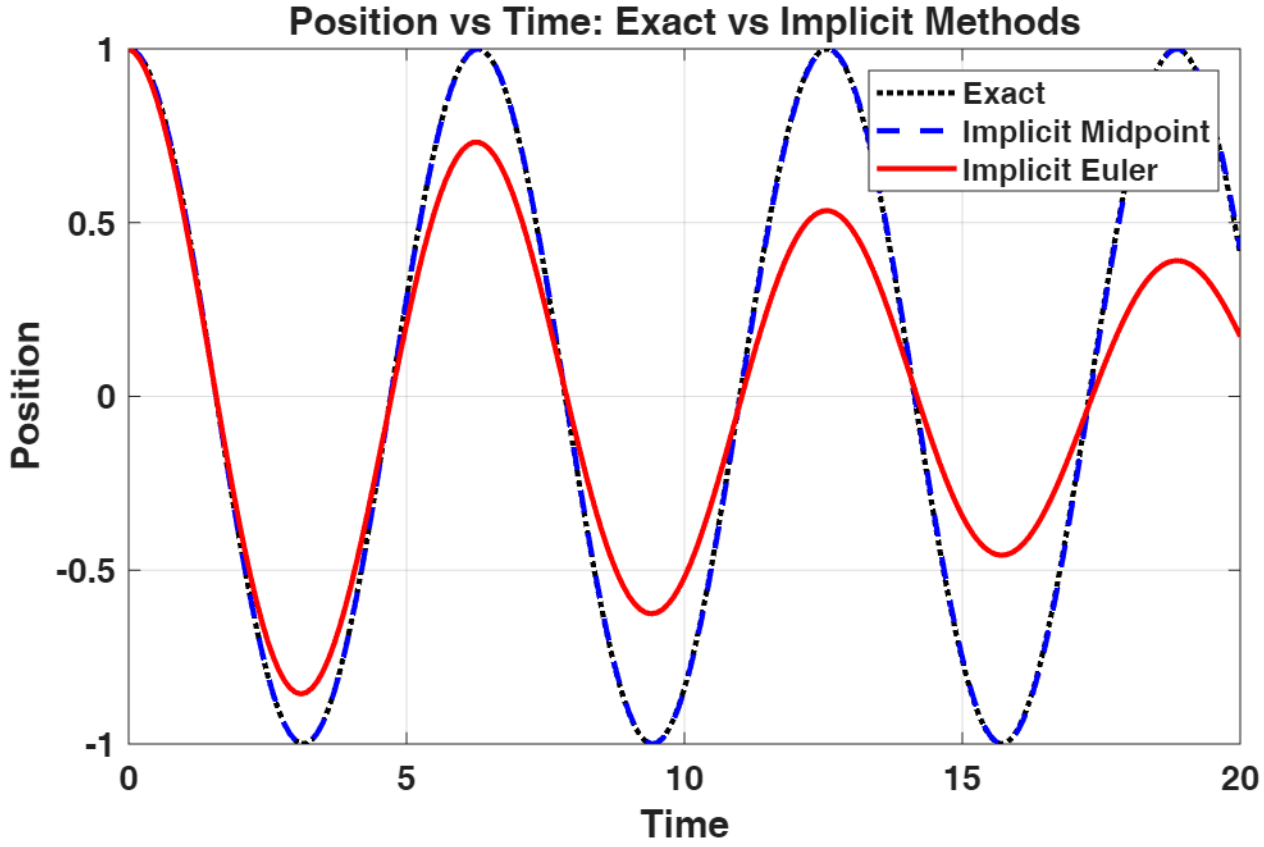
- Newton iteration: $y(k+1) = y(k) - R(y(k))/R'(y(k))$
- Efficient for nonlinear equations
- Fast (quadratic) convergence near solution
- Works for general nonlinear forces (plasma physics)

Implicit Midpoint Method: Concept

- Time-centered implicit integration method
- Evaluates system at midpoint of time step
- High accuracy for oscillatory systems
- Widely used in physics simulations
  - $dx/dt = v$, $dv/dt = a(x)$
  - $x(n+1) = x(n) + \Delta t\,(v(n) + v(n+1))/2$
  - $v(n+1) = v(n) + \Delta t\, a((x(n) + x(n+1))/2)$

Solve nonlinear system (Newton method)

Implicit Midpoint Vs Euler

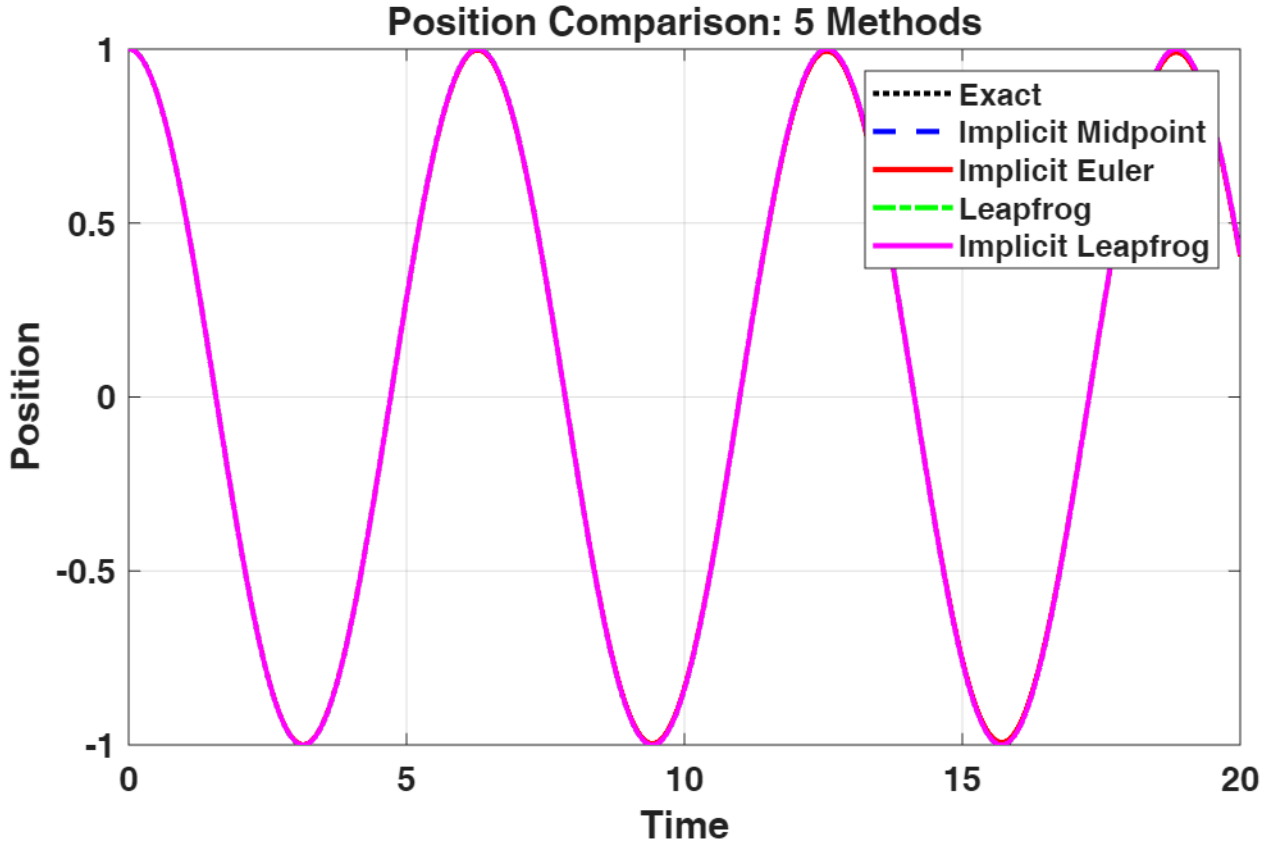


- dt=0.1
- Midpoint: Second-order accurate (O(Δt^2))
- Good energy conservation

Implicit Leapfrog Method: Concept

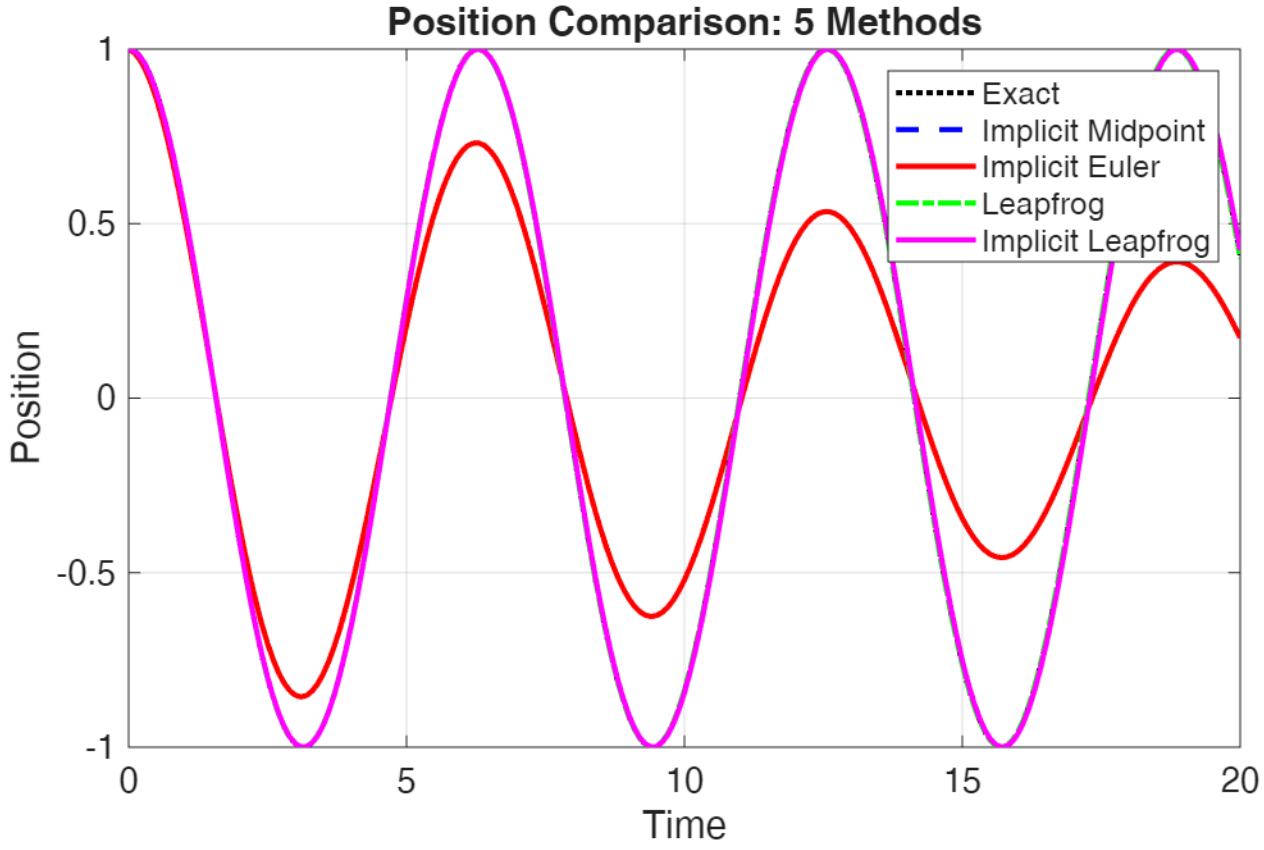
- Time-centered integration scheme, Force evaluated at midpoint position, Improves accuracy for oscillatory motion, Implicit because future position appears in force
- $V(n+1/2) = v(n) + (\Delta t/2)\, a((x(n) + x(n+1))/2)$
- $X(n+1) = x(n) + \Delta t\, v^{n+1/2}$
- $v^{n+1} = v(n) + \Delta t\, a((x(n) + x(n+1))/2)$
- Second-order accurate ($O(\Delta t^2)$) and More stable than explicit Euler

Comparison



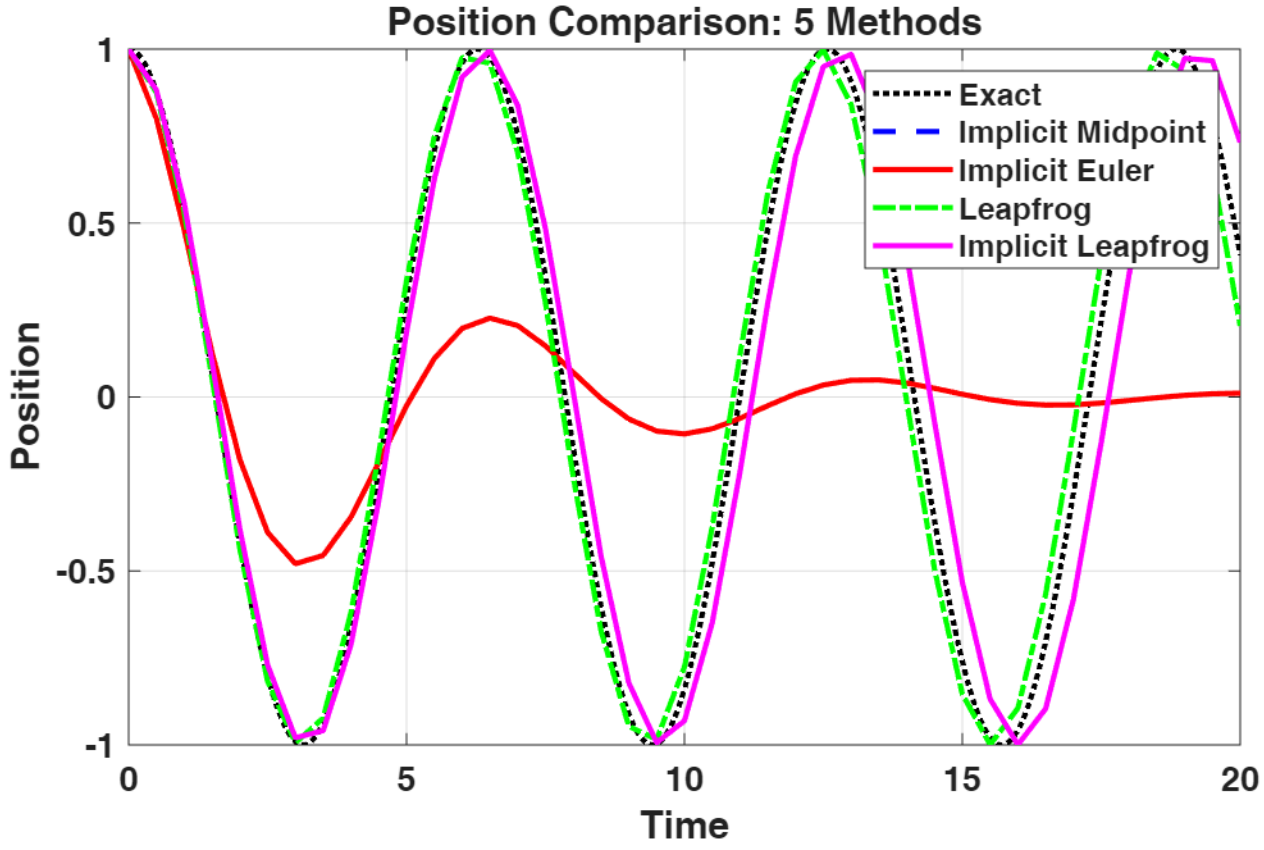
- dt=0.001

Comparison



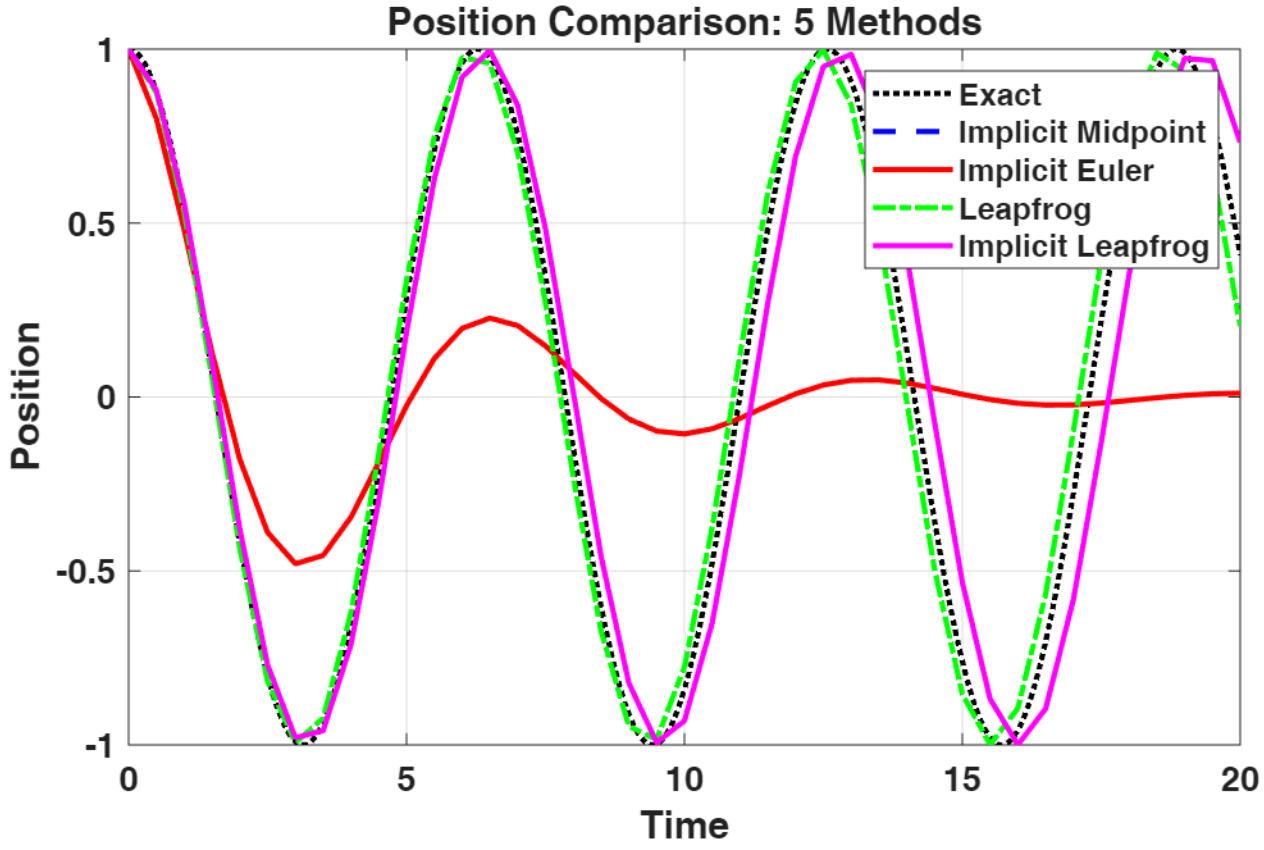
- dt=0.01

Comparison



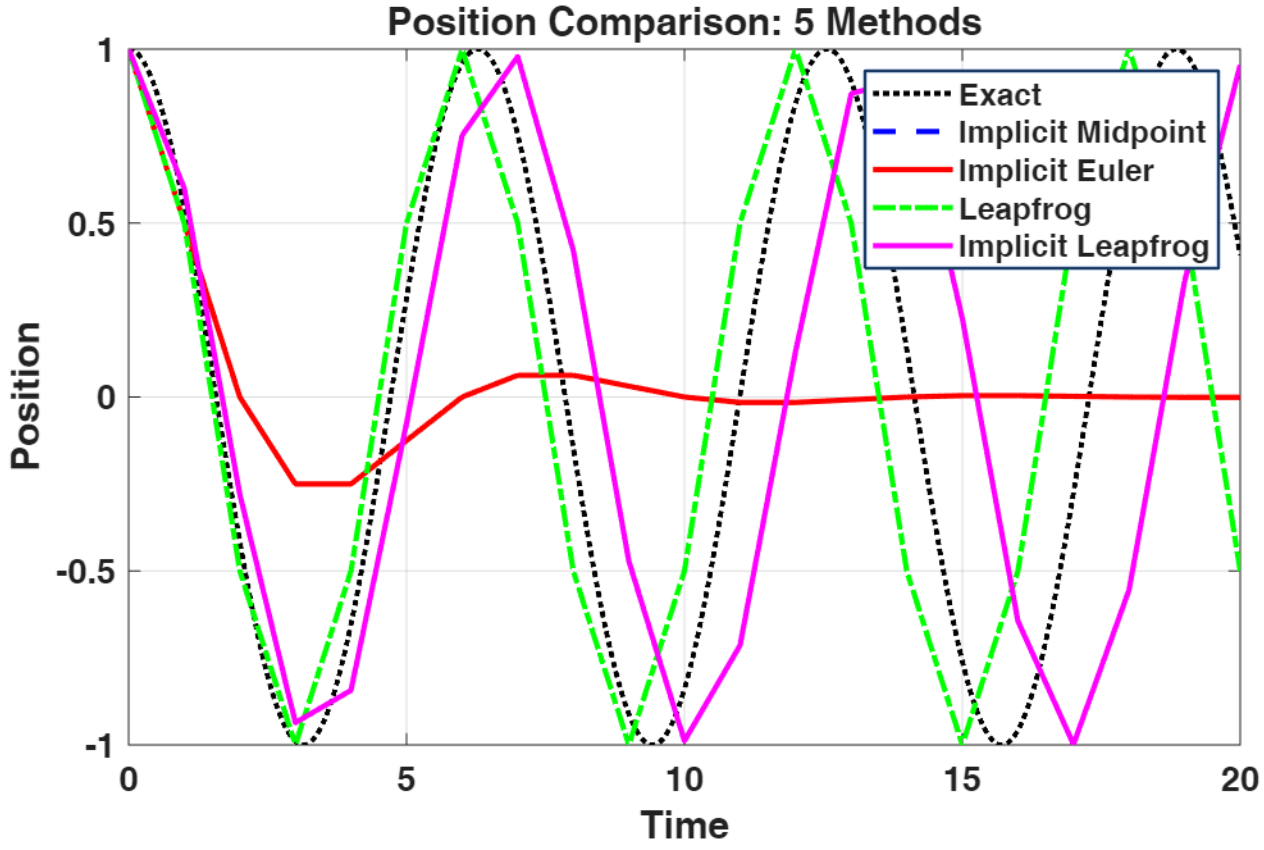
- dt=0.01

Comparison



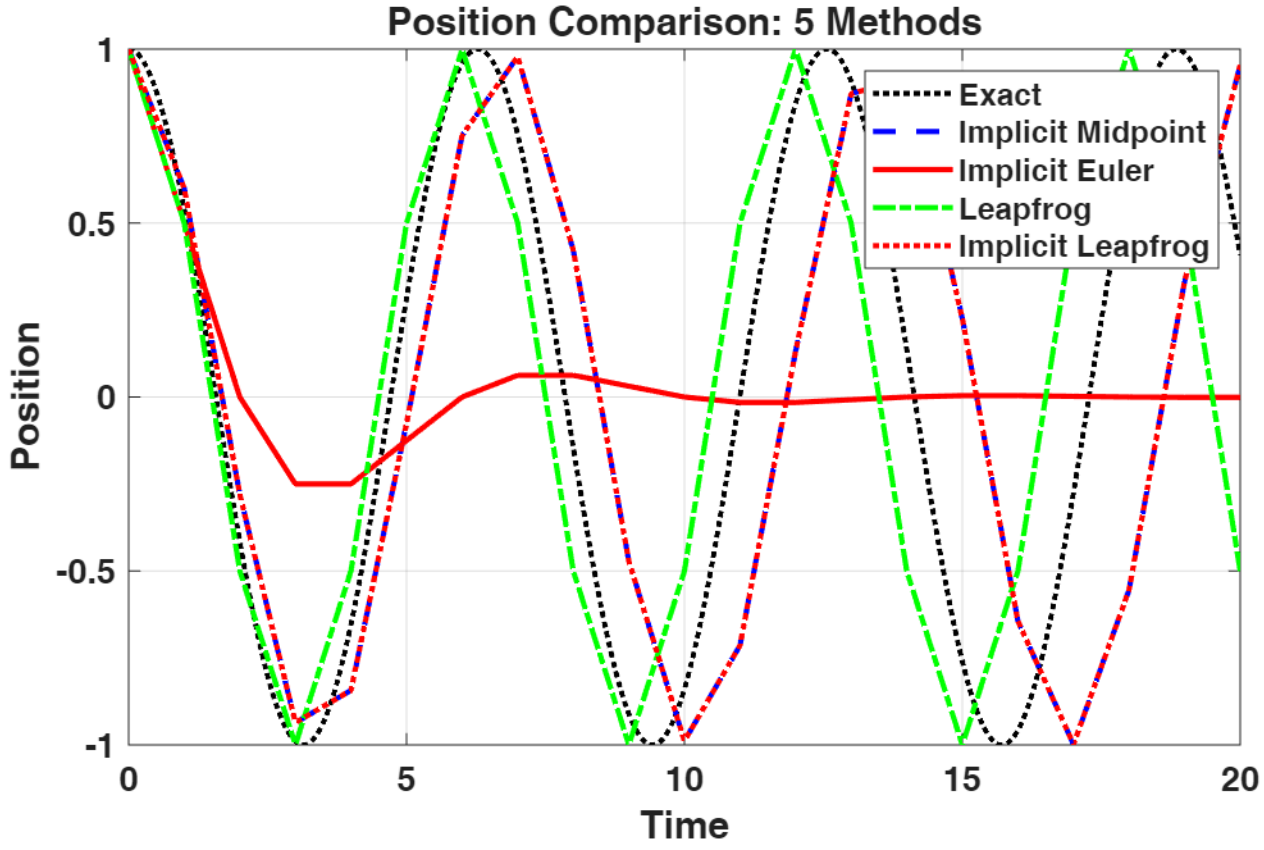
- dt=0.5

Comparison



- dt=1
  - We have to consider ω dt << 1

Comparison

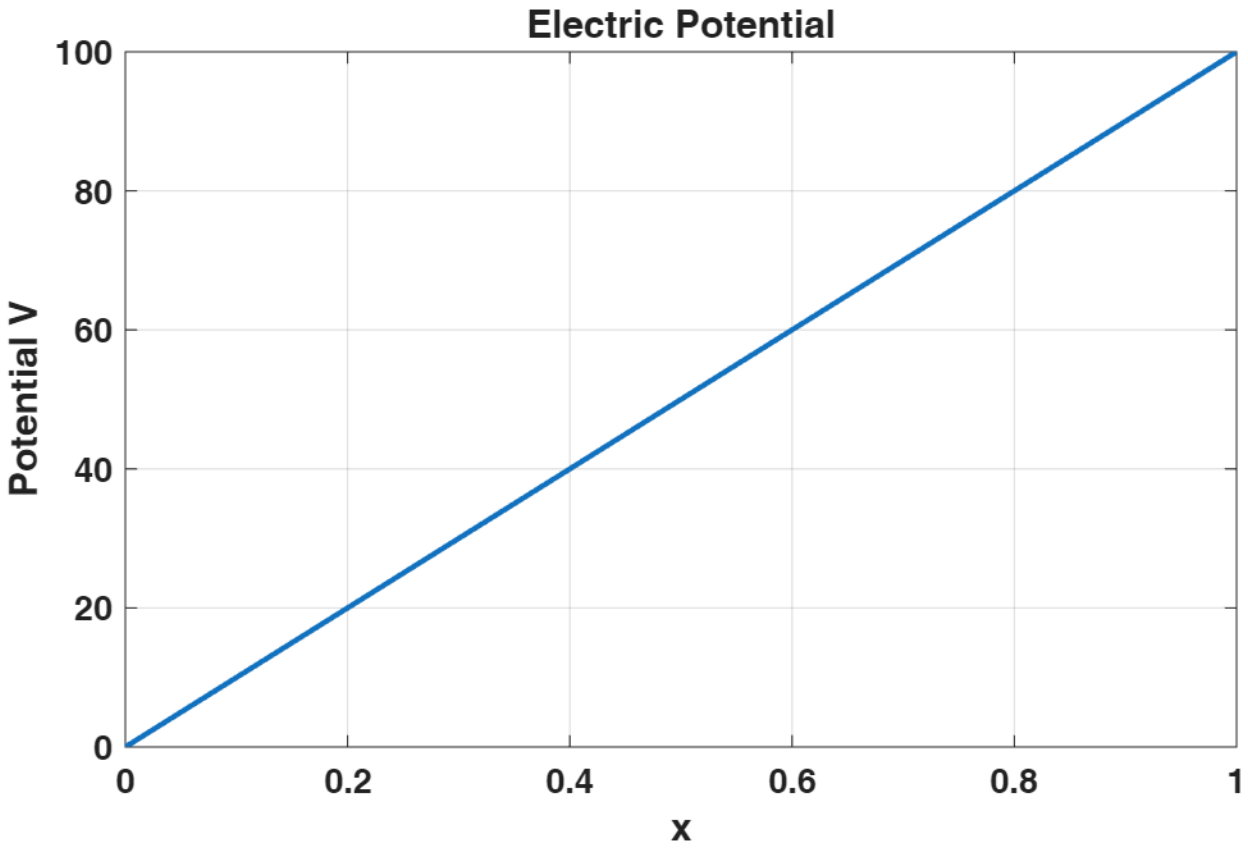


- dt=1
- We have to consider ω dt << 1
- Leapfrog is the best one.

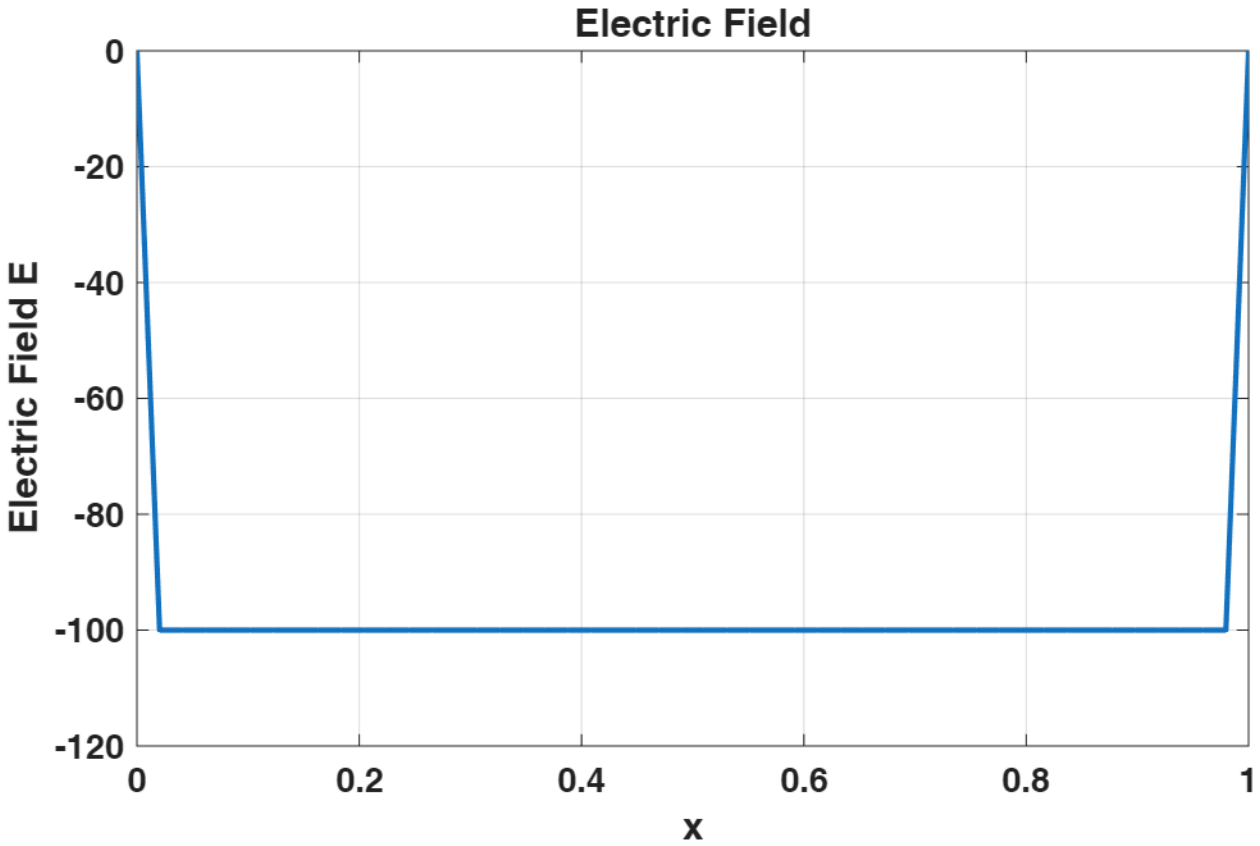
1D Electrostatic Problem

- Equation: $d^2V/dx^2 = 0$
- Exact solution: $V(x) = 100 * x / L$
- Electric field: $E = -dV/dx = \text{constant}$
- Finite difference: $V(i) = (V(i+1) + V(i-1)) / 2$

1D Electrostatic Problem



1D Electrostatic Problem

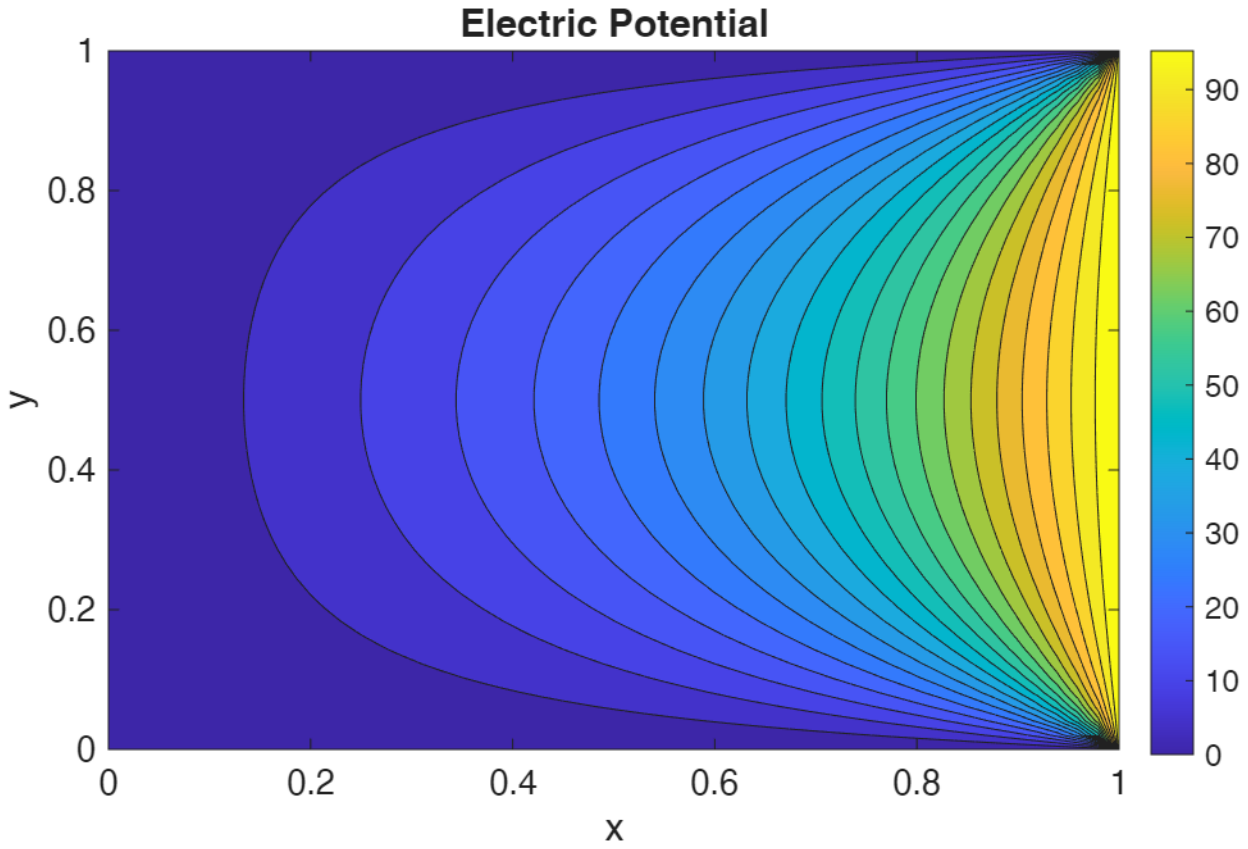


Electrostatic Problem Setup

Solve Laplace equation: $\nabla^2 V = 0$

- Two electrodes: 0 V (left), 100 V (right)
- Discretize Laplace equation:
- $V(i,j) = (V(i+1,j)+V(i-1,j)+V(i,j+1)+V(i,j-1))/4$
- Use Gauss-Seidel iteration, update grid until convergence (error < tolerance)
- Electric field: $E = -\nabla V$
- $Ex = -(V(i,j+1)-V(i,j-1))/(2dx)$
- $Ey = -(V(i+1,j)-V(i-1,j))/(2dy)$

2D



1D Poisson Equation: Implicit

- Solve: $d^2V/dx^2 = -\rho(x)/\varepsilon$
- V(x): electric potential, ρ(x): charge density
- Boundary conditions: V(0), V(L)
- Approximate second derivative:
- $(V(i+1) - 2V(i) + V(i-1)) / dx^2 = -\rho(i)/\varepsilon$
- Rearranged:
- $-V(i-1) + 2V(i) - V(i+1) = \rho(i)\, dx^2 / \varepsilon$
- One equation per grid point

Implicit Matrix Formulation

- System written as: A V = b
- A: tridiagonal matrix (couples all points)
- V: unknown potential vector
- b: includes charge density + boundary conditions
- Solve using: V = A \ b

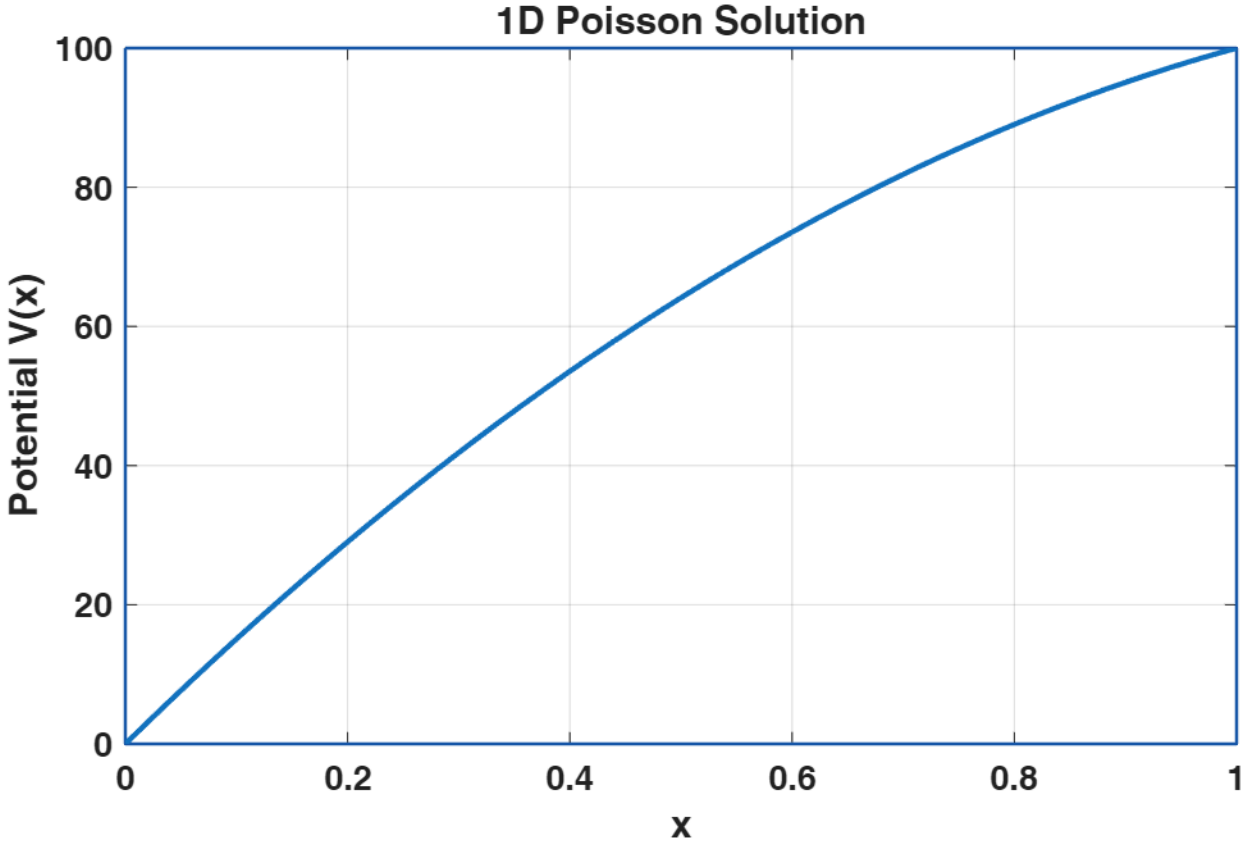
Electric Field & Physical Meaning

- Electric field: $E = -dV/dx$
- Numerically: $E(i) = -(V(i+1)-V(i-1))/(2dx)$
- $\rho = 0 \rightarrow$ linear potential, constant field
- $\rho \neq 0 \rightarrow$ curved potential, varying field
- Implicit method solves entire system at once

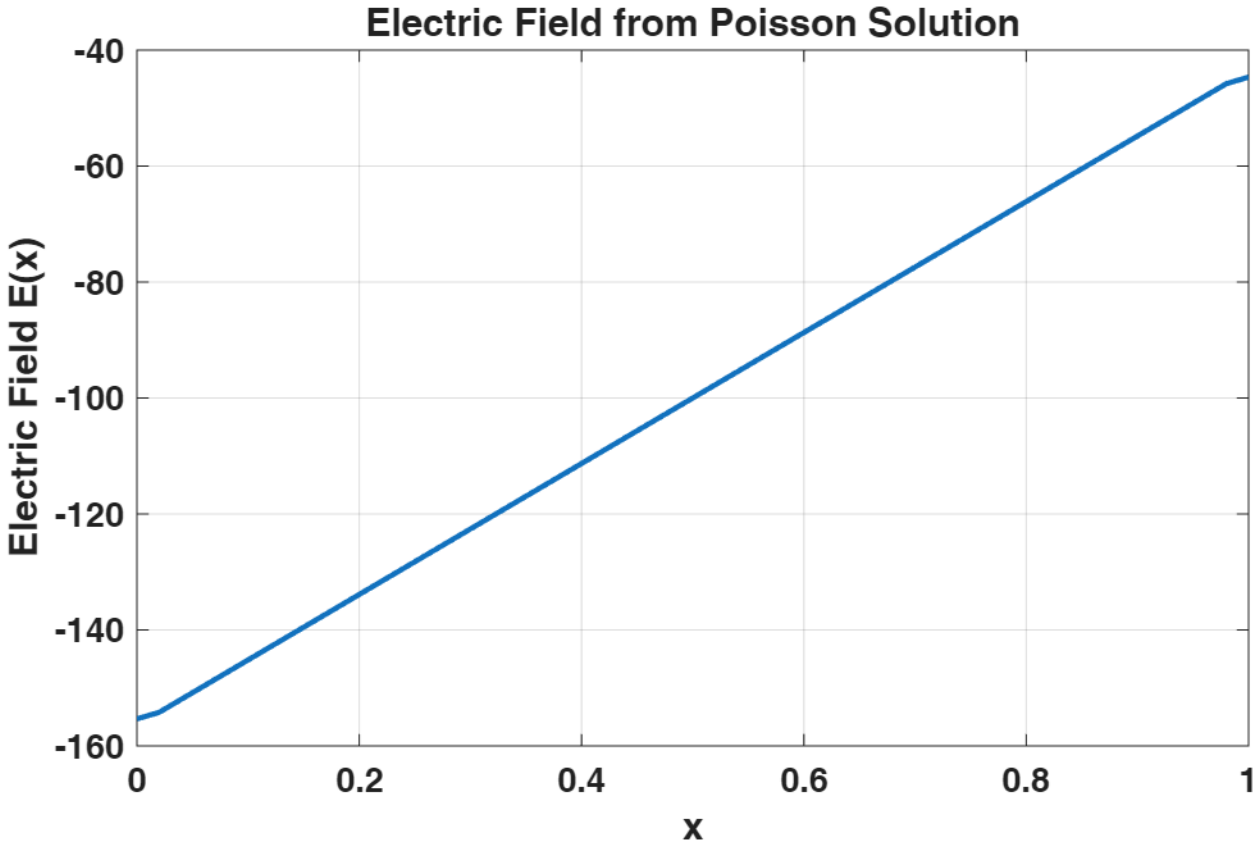
Electric Field & Physical Meaning

- Electric field: $E = -dV/dx$
- Numerically: $E(i) = -(V(i+1)-V(i-1))/(2dx)$
- $\rho = 0 \rightarrow$ linear potential, constant field
- $\rho \neq 0 \rightarrow$ curved potential, varying field
- Implicit method solves entire system at once

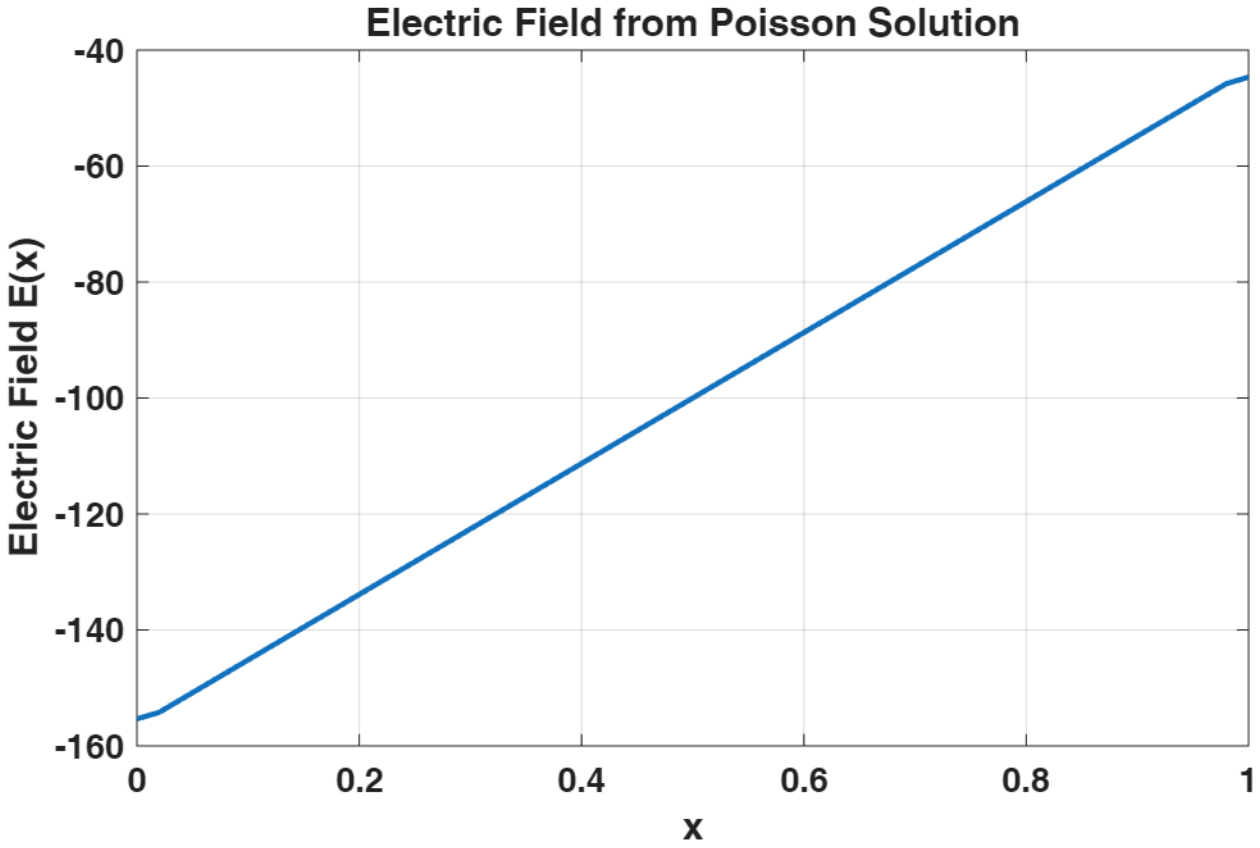
Electric Field & Physical Meaning



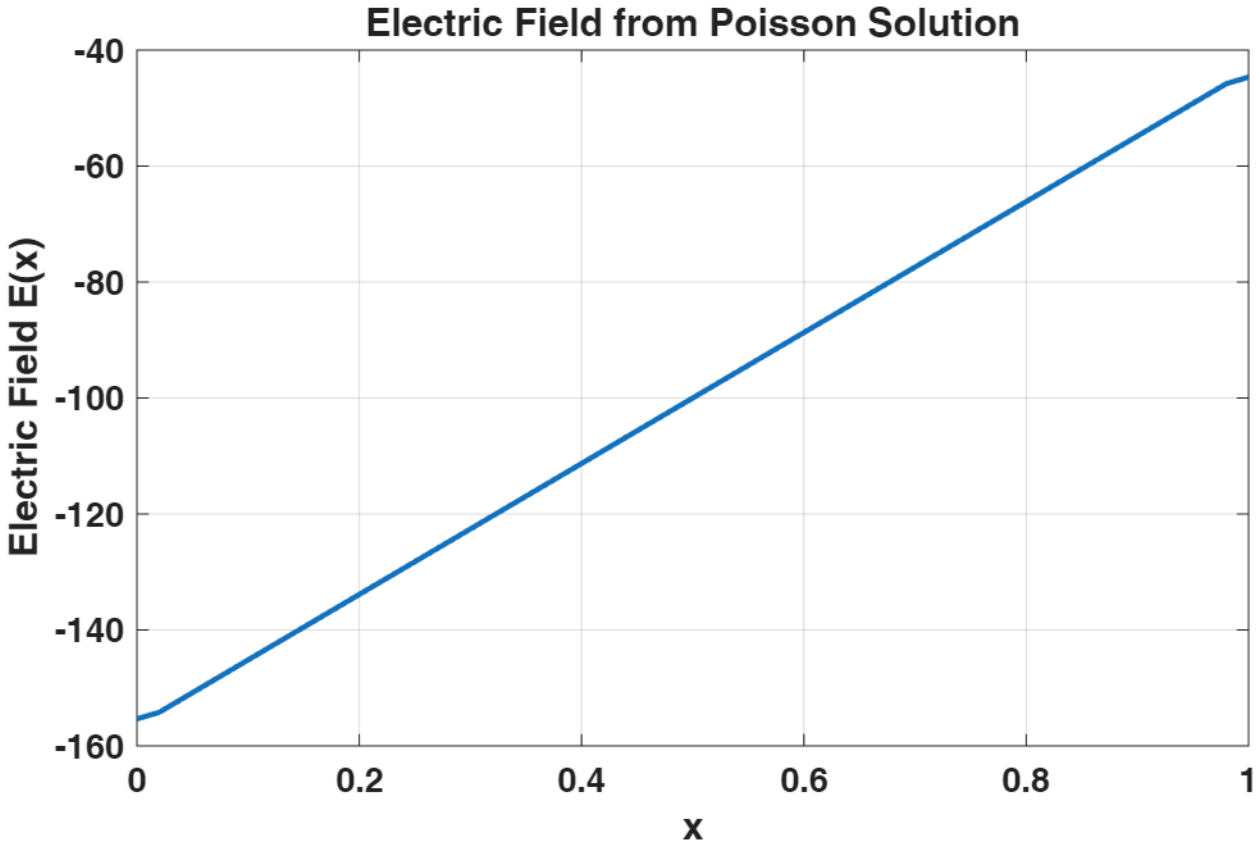
Electric Field & Physical Meaning



Electric Field & Physical Meaning



Electric Field & Physical Meaning



1D Poisson Equation (Fourier Method)

- Solve: $d^2\varphi/dx^2 = -\rho(x)/\varepsilon$
- φ: potential, ρ: charge density
- Goal: compute φ and electric field E
- Assume periodic boundary conditions

Key Idea of Fourier Method

- Transform equation into Fourier space
- Derivative becomes algebraic operator
- $d/dx \rightarrow ik$, $d^2/dx^2 \rightarrow -k^2$
- Convert differential equation $\rightarrow$ algebraic equation. $\varphi(x) \leftrightarrow \hat{\varphi}(k)$, $\rho(x) \leftrightarrow \hat{\rho}(k)$
- Apply FFT to transform $\rho(x)$
- Poisson equation becomes:
- $-k^2 \hat{\varphi}(k) = -\hat{\rho}(k)/\varepsilon$

Algorithm Steps

- 1. Define $\rho(x)$ on grid
- 2. Compute FFT $\rightarrow \hat{\rho}(k)$
- 3. Solve $\hat{\varphi}(k) = \hat{\rho}(k)/(\varepsilon\, k^2)$
- 4. Inverse FFT $\rightarrow \varphi(x)$
- 5. Compute $E(x)$ from φ

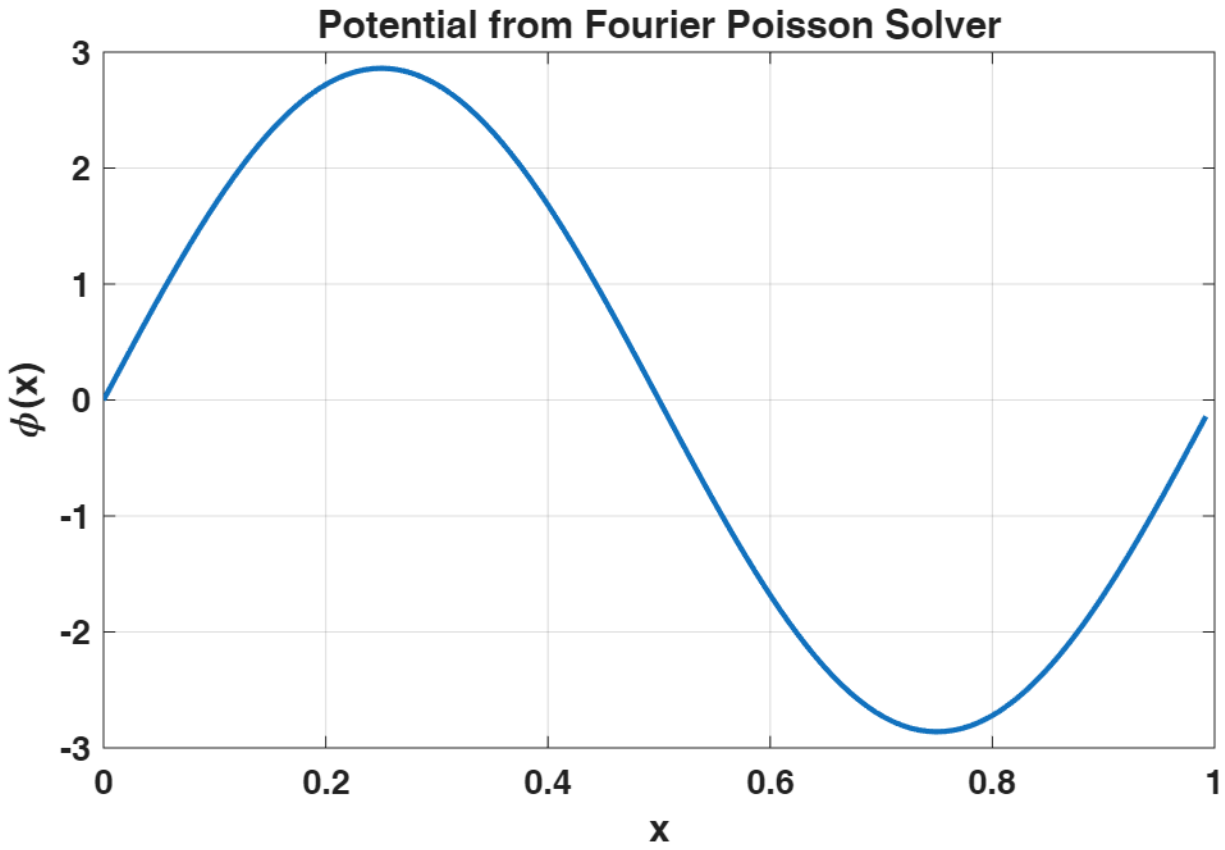
Advantages and Limitations

- Advantages: Highly accurate for smooth periodic functions, Fast: O(N log N) using FFT, No iteration required, Handles derivatives naturally
- Limitations: Requires periodic boundary conditions, k = 0 mode requires special treatment, Not suitable for fixed boundary potentials directly

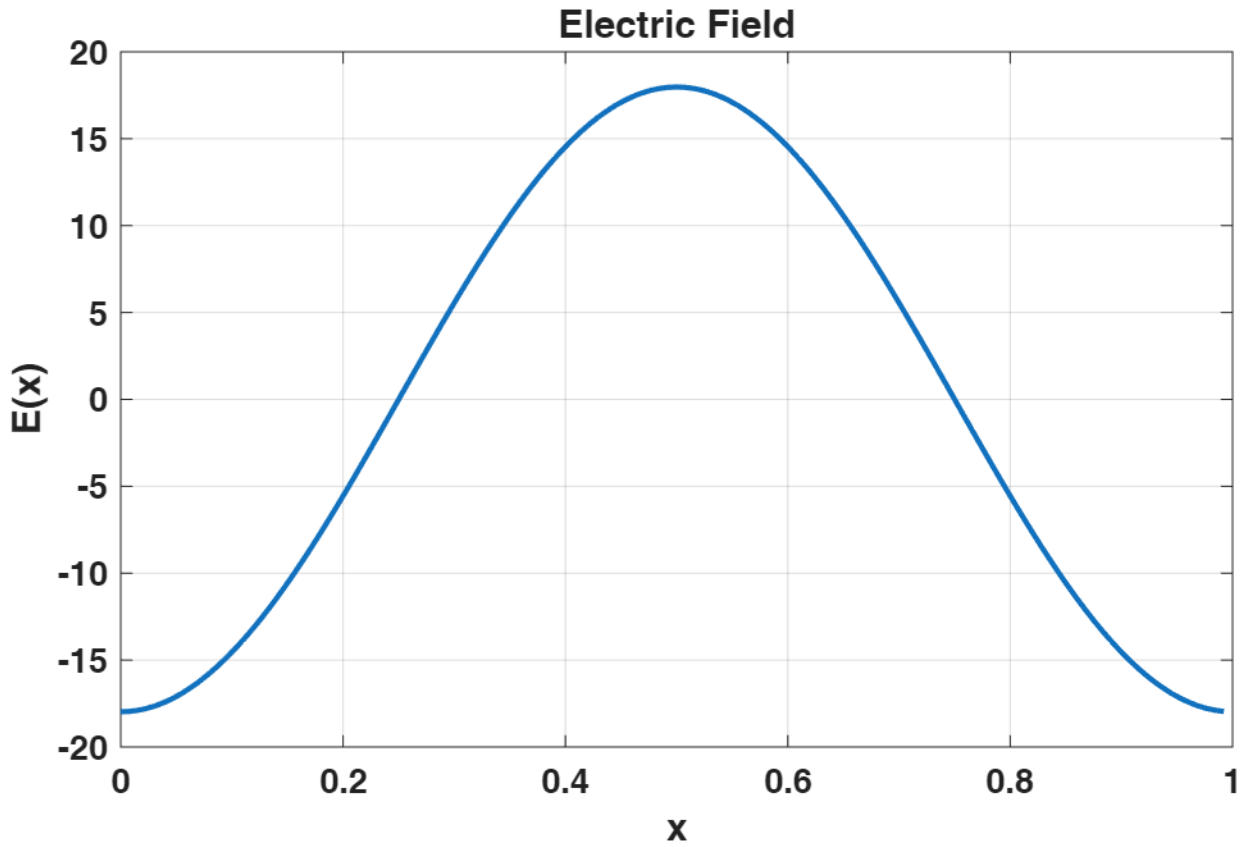
Physical Interpretation

- Each Fourier mode represents a wave
- Poisson equation solved mode-by-mode
- Long wavelength modes → larger potential
- Electric field derived from spatial variation

Potential



Field



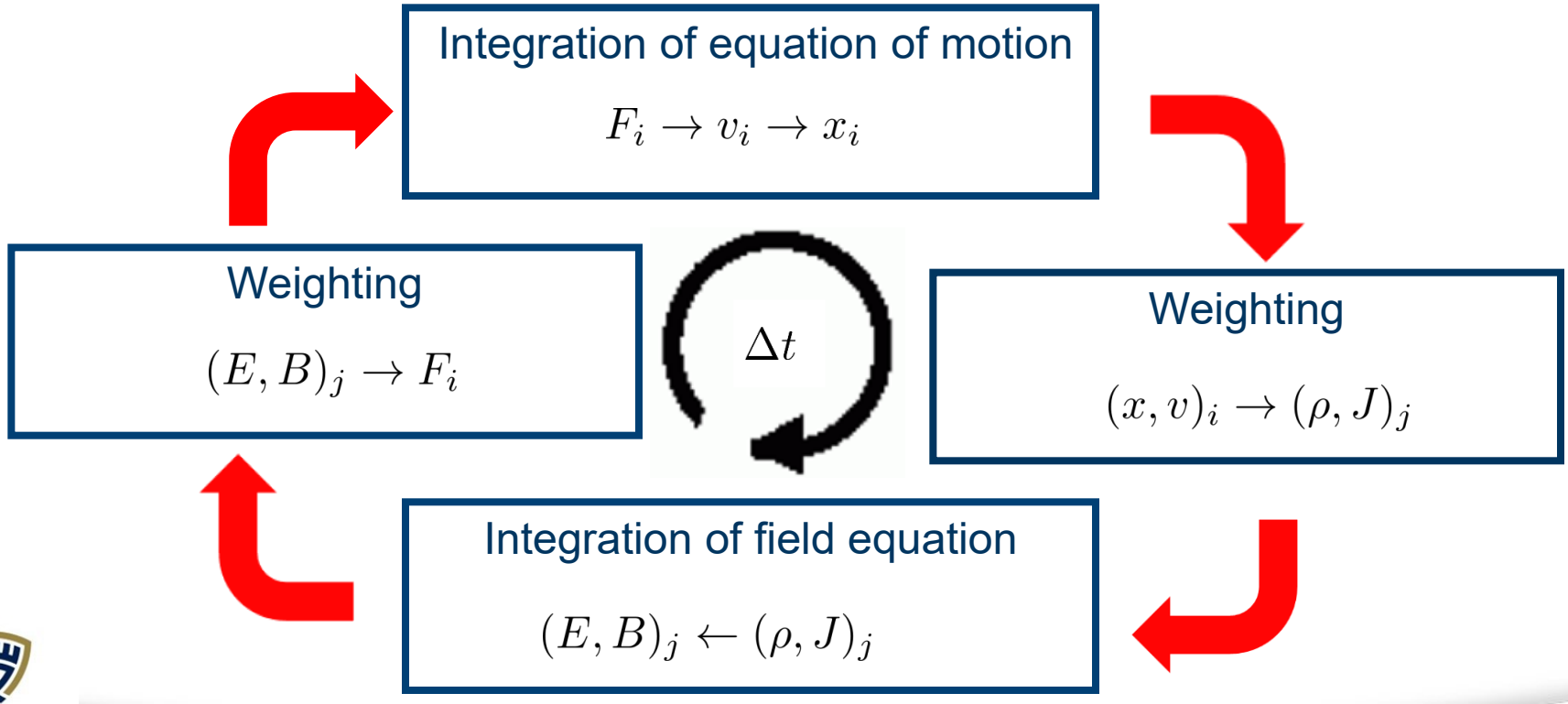
Thank You!

We are ready to start PIC simulation

جامعة السلام
AL SALAM UNIVERSITY

Particle in Cell Simulation

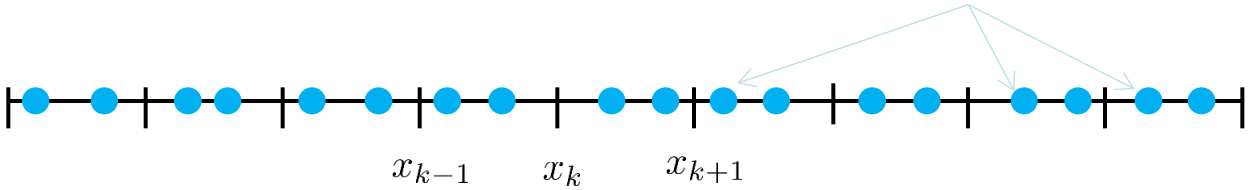
- Kinetic means „of or relating to motion“.
  - It is impractical to solve the equation of motion of all plasma particles.
  - Boltzman equation is an integro-differential equation.
- Particle-in-Cell : Super particle $10^6 - 10^9$ real particles.



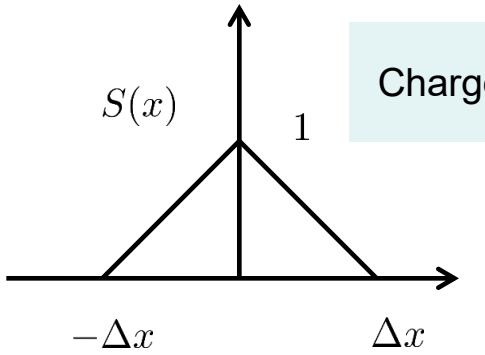
Particles

1D Problem

Super-particles

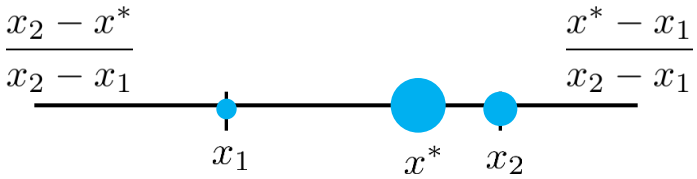


Simulation domain



Charge cloud

First order wieghting



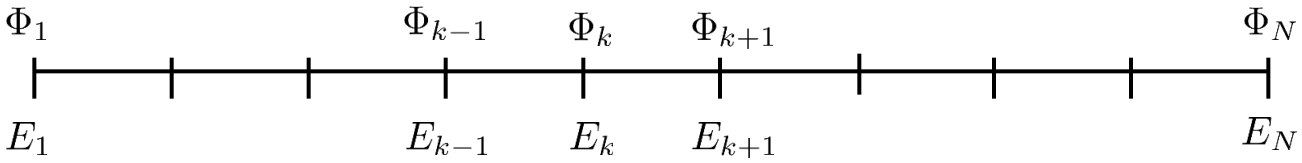
Fields

Poisson's eq.

$$\nabla^2\Phi = -\frac{\rho}{\epsilon} \quad \rightarrow \quad \frac{\Phi_{k+1} - 2\Phi_k + \Phi_{k-1}}{\Delta x^2} = -\frac{\rho}{\epsilon}$$

Boundary condition

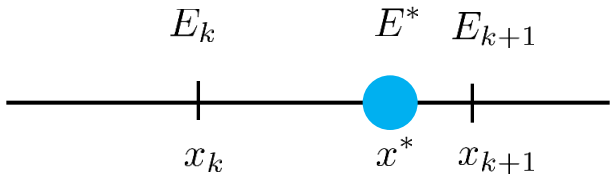
Boundary condition



$$E_1 = -\frac{\Phi_2 - \Phi_1}{\Delta x} - \rho_1 \frac{\Delta x}{2\epsilon} \qquad E_k = -\frac{\Phi_{k+1} - \Phi_{k-1}}{2\Delta x} \qquad E_N = -\frac{\Phi_N - \Phi_{N-1}}{\Delta x} + \rho_N \frac{\Delta x}{2\epsilon}$$

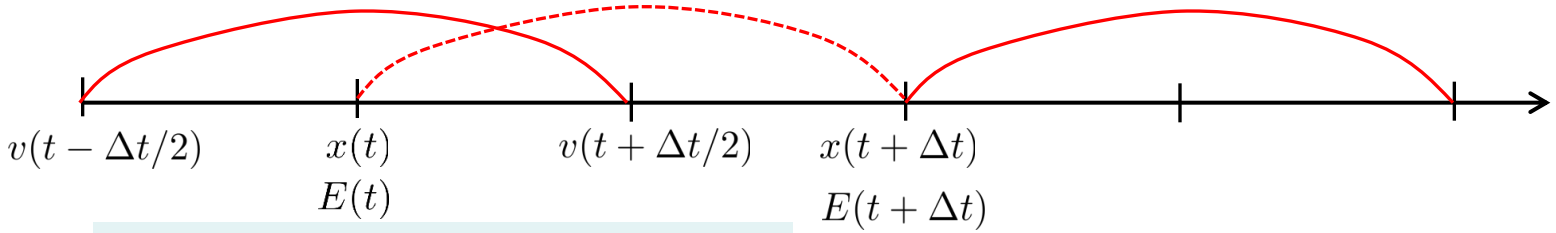
Interpolation of the fields to the particle positions

$$E^* = \frac{x^* - x_k}{\Delta x} E_{k+1} + \frac{x_{k+1} - x^*}{\Delta x} E_k$$



Pushing particles

„Leapfrog“ scheme



Descritization of equation of motions

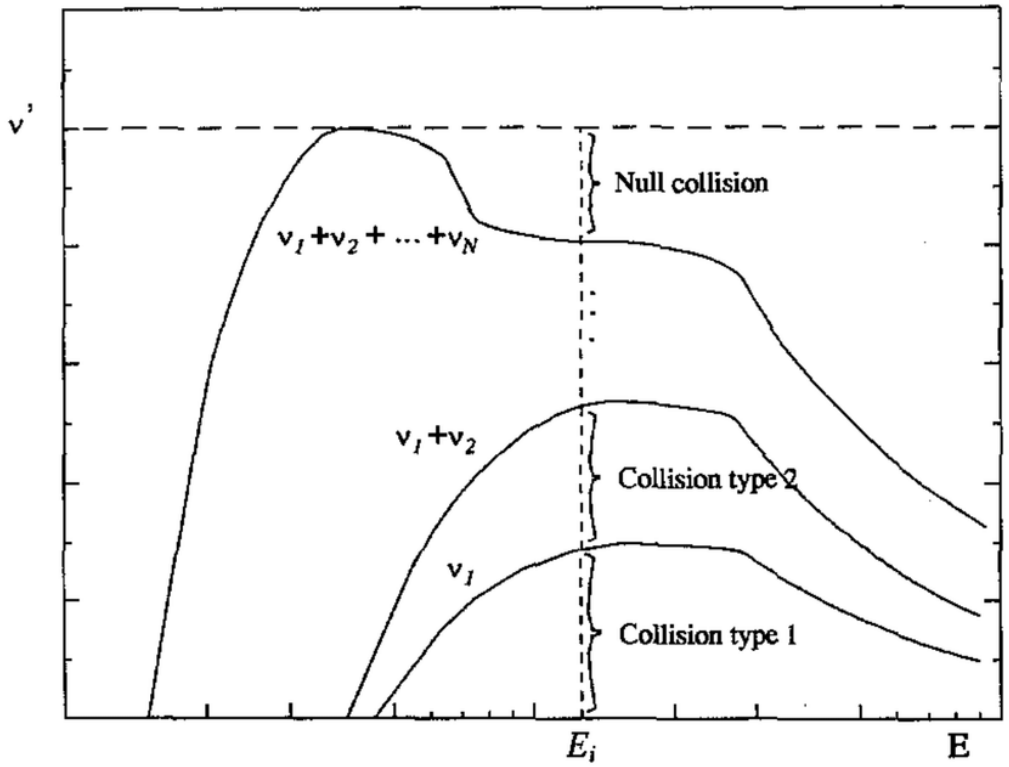
$$\frac{v(t+\Delta t/2)-v(t-\Delta t/2)}{\Delta t}=\frac{q}{m}E(t)$$

$$\frac{x(t+\Delta t)-x(t)}{\Delta t}=v(t+\Delta t/2)$$

Monte-Carlo Scheme is required for collisions

Monte Carlo: null collision method

- Many collisions take place:
  impact ionization, charge exchange,
  hard-shere, ...
- Let the probability of them
  P1,P2,P3,P4, ...
- Claculate the total probabilities PT
- Calculate relative probabilities
  P1/PT, P2/PT, P3/PT, P4/PT, ...



- Generate a random number between [0,1]
- if P1/PT= < The randum number<(P1+P2)/PT
- Event 1 takes place
- If not

(P1+ P2)/PT= < The randum number<(P1+P2+P3)/PT

Challenges of PIC simulation

- Numerical instabilities:
  - Accuracy criterion $\omega_p \Delta t \leq 0.2$
  - Courant criterion $v_{\max} \Delta t \leq \Delta x$
  - The computational grid has to resolve the Debye lenght $\Delta x \leq \lambda_D$
- In order to have a good statistics, a resonable high number of particles per Debye lenght must be used $N_{\mathrm{D}} \gg 1$
- Keep the probabilty for collisions small

$$P_{\mathrm{coll}} = 1 - e^{-\nu t} \leq 0.1$$

- Alternatives:
  - Implicit schems
  - Parrallilization