Agenda

- Introduction to paging and paging delay
- Problem statement and objectives
- Mathematical modelling using LDE
- Algorithm design and implementation
- Program Implementation
- Validation & Simulation
- Results & Conclusion
- Future scope
- RAM Doctor (a web page)
- Conclusion

Introduction

- **Efficient memory management** is crucial for high performance in modern computer systems, especially when multiple programs compete for limited physical memory.
- **Virtual memory** provides an abstraction that allows processes to use more memory than physically available by storing inactive pages on secondary storage.
- **Paging** is a key mechanism in virtual memory:
  - Memory is divided into fixed-size pages.
  - Pages are swapped between main memory and disk based on process demand.
- **Congestion in memory queues** occurs when:
  - Incoming memory requests increase.
  - Multiple processes frequently access memory pages.
- **Paging delay**:
  - Represents the time a process waits for its required page to become available.
  - Directly impacts system throughput, CPU utilization, and user experience in multitasking OS.
- **Modeling paging delay**:
  - Provides insights into system responsiveness.
  - Helps design efficient memory management strategies.

Problem statement

In a virtual memory system, processes often have to wait for memory pages due to page faults or heavy memory load.

This waiting time is called paging delay. The behavior of this delay can be represented using a differential equation.

The memory manager gives the paging delay w(t) through a second-order differential equation.

In this equation:

$$\frac{d^2w(t)}{dt^2} + 5\frac{dw(t)}{dt} + 6w(t) = 10e^{-2t}$$

where,

w(t) = paging delay in seconds per request

$10e^{-2t}$ = the external paging load acting on the system

Objectives

The task is to:

Solve the given differential equation to find the exact expression for the paging delay w(t).

Use MATLAB/Python to compute and plot how w(t) changes over time.

Analyze how the system behaves—whether the delay increases, decreases, or stabilizes under the given load.

In short, the problem is about mathematically modeling paging delay, solving the equation, and understanding how memory load affects system performance.

Mathematical modelling

1. *Complementary Function (Homogeneous LDE)*

Consider the homogeneous differential equation

$$\frac{d^2w}{dt^2} + 5\frac{dw}{dt} + 6w = 0$$

To solve this, we form the auxiliary equation:

$$m^2 + 5m + 6 = 0$$

Factorizing:

$$(m + 2)(m + 3) = 0$$

Therefore, the roots are:

$$m = -2 , m = -3$$

Therefore, the complementary function is

$$w_c(t) = C_1e^{-2t} + C_2e^{-3t}$$

2. *Particular Integral (Non-homogeneous Part*

We now solve the non-homogeneous equation:

$$\frac{d^2w}{dt^2} + 5\frac{dw}{dt} + 6w = 10e^{-2t}$$

In operator form:

$$(D^2 + 5D + 6)w = 10e^{-2t}$$

Attempting the usual form:

$$w_p = \frac{10e^{-2t}}{D^2 + 5D + 6}$$

Substituting D = −2D

$$(-2)^2 + 5(-2) + 6 = 0$$

Since the denominator becomes zero, the term e^{-2t} is a root of the auxiliary equation. Therefore, we multiply by t and use the derivative of the operator. Let

$$f(m) = m^2 + 5m + 6$$

Then: *f'(m) =2m+5f'(m)*

Evaluate at m = −2m

f'(−2) = 2(−2) + 5 = 1

Thus, the particular integral is: $w_p(t) = 10te^{-2t}$

3. *General Solution of the Differential Equation*

The complete solution is obtained by adding the complementary function and particular integral:

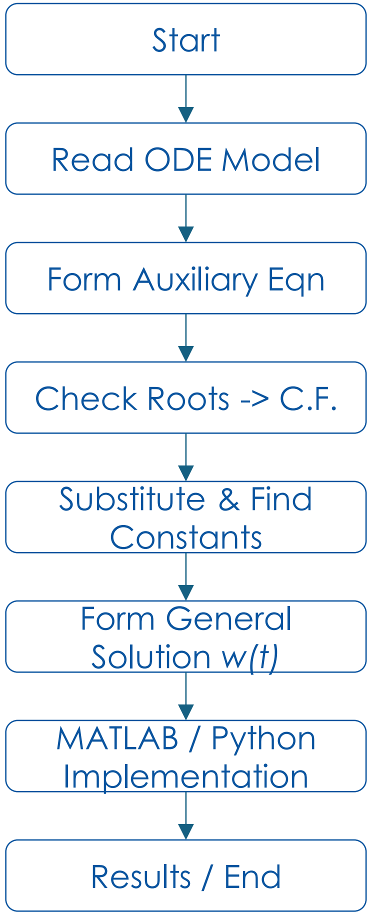
$$w(t) = C_1e^{-2t} + C_2e^{-3t} + 10te^{-2t}$$

Algorithm design and implementation

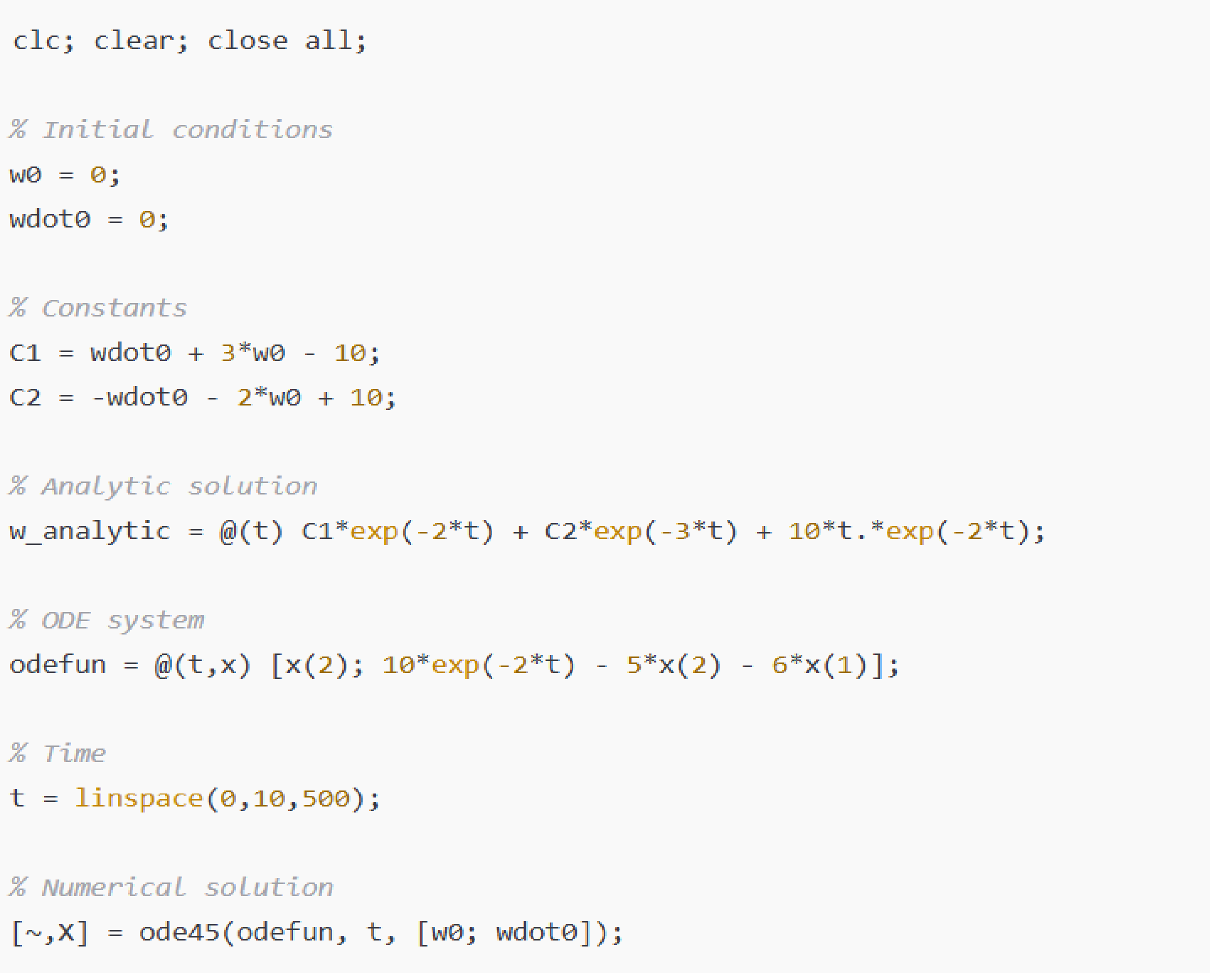
Algorithm Design

1. Start the process.
2. Read the paging delay differential equation.
3. Form the auxiliary equation from the homogeneous part.
4. Solve for roots → obtain complementary function.
5. Check RHS term; if it matches a root, modify the particular solution by multiplying with t.
6. Substitute trial solution into the ODE and find constants.
7. Form the general solution w(t) = wc(t) + wp(t) .
8. Implement the solution using MATLAB/Python.
9. Plot the result and analyze the paging delay behavior.
10. End.

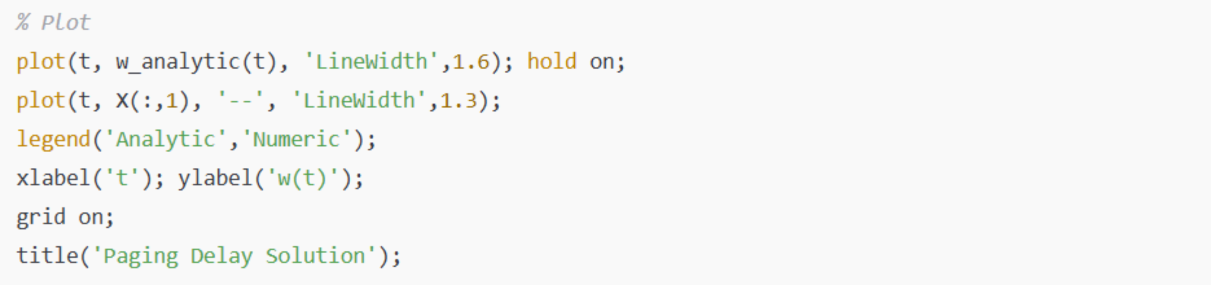
Schematic Diagram


Start
Read ODE Model
Form Auxiliary Eqn
Check Roots -> C.F.
Substitute & Find Constants
Form General Solution *w(t)*
MATLAB / Python Implementation
Results / End

Program Implementation



```
clc; clear; close all;

% Initial conditions
w0 = 0;
wdot0 = 0;

% Constants
C1 = wdot0 + 3*w0 - 10;
C2 = -wdot0 - 2*w0 + 10;

% Analytic solution
w_analytic = @(t) C1*exp(-2*t) + C2*exp(-3*t) + 10*t.*exp(-2*t);

% ODE system
odefun = @(t,x) [x(2); 10*exp(-2*t) - 5*x(2) - 6*x(1)];

% Time
t = linspace(0,10,500);

% Numerical solution
[~,X] = ode45(odefun, t, [w0; wdot0]);
```



```
% Plot
plot(t, w_analytic(t), 'LineWidth',1.6); hold on;
plot(t, X(:,1), '--', 'LineWidth',1.3);
legend('Analytic','Numeric');
xlabel('t'); ylabel('w(t)');
grid on;
title('Paging Delay Solution');
```

Validation & Simulation

1. ***Analytical Solution***

   The derived closed-form solution is:

   $$w(t) = C_1 e^{-2t} + C_2 e^{-3t} + 10te^{-2t}$$

2. ***Numerical Simulation***

   The same differential equation is solved numerically using MATLAB's ode45, which is a highly accurate Runge–Kutta method.

3. ***Comparison***

   - Both solutions are plotted on the same graph.
   - The curves overlap almost perfectly, showing that the analytical solution is correct.
   - The absolute error:

   $$w_{\text{num}}(t) - w_{\text{analytic}}(t)| \approx 10^{-8} \text{ to } 10^{-12}$$

Results & Conclusion

1. The differential equation

$$w''(t) + 5w'(t) + 6w(t) = 10e^{-2t}$$

was solved analytically to obtain the general solution:

$$w(t) = C_1e^{-2t} + C_2e^{-3t} + 10te^{-2t}$$

1. MATLAB simulation (ode45), the numerical solution was computed over the range 0 ≤ t ≤ 10.
2. When the analytical and numerical results were plotted together, both curves overlapped almost perfectly.
3. The validation error

   $$|w_{\text{numeric}} - w_{\text{analytic}}| \approx 10^{-8} \text{ to } 10^{-12}$$

   indicates extremely high accuracy of the analytical solution.
4. The paging delay initially increases due to the exponential input load and later decreases exponentially, showing stable system behavior.

Future scope

- **Advanced Numerical Methods:**

  More accurate or faster numerical solvers (such as higher-order Runge–Kutta or adaptive multi-step methods) can be used to analyze the paging delay under different operating conditions.

- **Real-System Data Integration:**

  The model can be improved by incorporating real operating system metrics such as page-fault rate, memory access patterns, and workload behavior.

- **Extended Models:**

  The differential equation can be expanded to include multi-level paging, TLB delays, or dynamic memory allocation techniques.

- **Stochastic Modeling:**

  Paging delays can be modeled using probabilistic or queueing models to represent random workloads or unpredictable system events.

- **Performance Optimization**:

  The model can be used to design and test memory management algorithms that minimize paging delay and improve system performance.

- **Machine Learning Integration:**

  Predictive models can be developed using machine learning to forecast paging delays and automatically optimize memory usage.

RAM Doctor (a web page)

RAM Doctor is an interactive simulation tool designed to help users understand how paging affects a computer’s performance when RAM becomes overloaded. In modern systems, multiple applications run at the same time, and when the available RAM is not enough, the operating system begins paging data to and from secondary storage. This causes paging delay, which leads to slow response, lag, and system instability. RAM Doctor visualizes this delay through a mathematical model and graphs, allowing users to see how system performance decreases as memory load increases.

Paging delay is modelled using a differential equation:

$$w(t) + 5w(t) + 6w(t) = L \cdot t \cdot e^{-2t}$$

The solution tells how delay w(t) behaves over time. By increasing L (memory load), the paging delay becomes higher — meaning the system becomes slower.

RAM Doctor (a web page)

History and Evolution of Paging

Paging began in the 1960s to overcome small RAM sizes.

Operating systems like Unix, Windows, Linux, macOS, Android, iOS still use paging today because apps need more memory than RAM alone can provide.

Why was paging introduced?

- Paging solved early memory problems:
- Programs were bigger than RAM
- Fixed partitions caused wastage
- Multitasking needed flexible memory usage
- Only the needed parts of a program stay in RAM → better memory utilization.

Disadvantages of Excessive Paging

Too much paging causes:

- Delay in process execution
- CPU waiting → lower performance
- App freezes, lag, “Not Responding”
- More battery usage and heat

This slowdown is called paging delay.

How to Use the Simulator

- Choose a memory load or set value of L
- Set simulation time
- Click Simulate Paging Delay
- Watch graph & table
- Compare different L values

Conclusion

RAM Doctor effectively connects theoretical concepts of memory management with real-world system performance. By simulating paging delay through a mathematical model, it shows how increasing memory load directly impacts the speed and responsiveness of a computer. The tool helps users visualize why devices slow down when RAM is full and highlights the importance of efficient memory usage. Overall, RAM Doctor serves as a useful educational tool for understanding paging, system behaviour, and the relationship between hardware limitations and performance.