Part 6

Ordinary Differential Equations

Chapter 22

Initial Value Problem

Ordinary Differential Equations

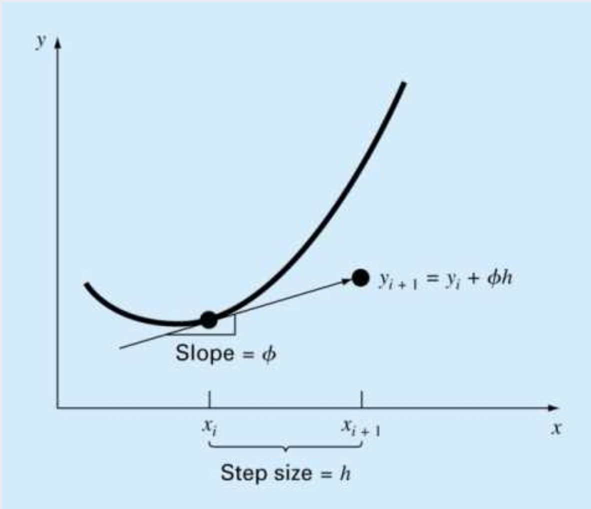
- Methods described here are for solving differential equations of the form : 시간과 공간에 대한 변화율 문제

$$\frac{dy}{dt} = f(t, y)$$

- *All one-step* methods have the general format:

$$y_{i+1} = y_i + \phi h$$

  - where ϕ is called an *increment function (slope)*, and is used to extrapolate from an old value y_i to a new value y_{i+1}.



Ordinary Differential Equation

- ODE initial value problem : $y(t_0)=y_0,\ y(t)=?$
  - One-step method
    - Euler's method
    - Heun's method
    - midpoint method
    - 4th order RK method

$$y_{i+1} = y_i + \phi h$$

  - Multistep method
    - Adaptive Runge-Kutta method

Euler's (or Euler-Cauchy) Method

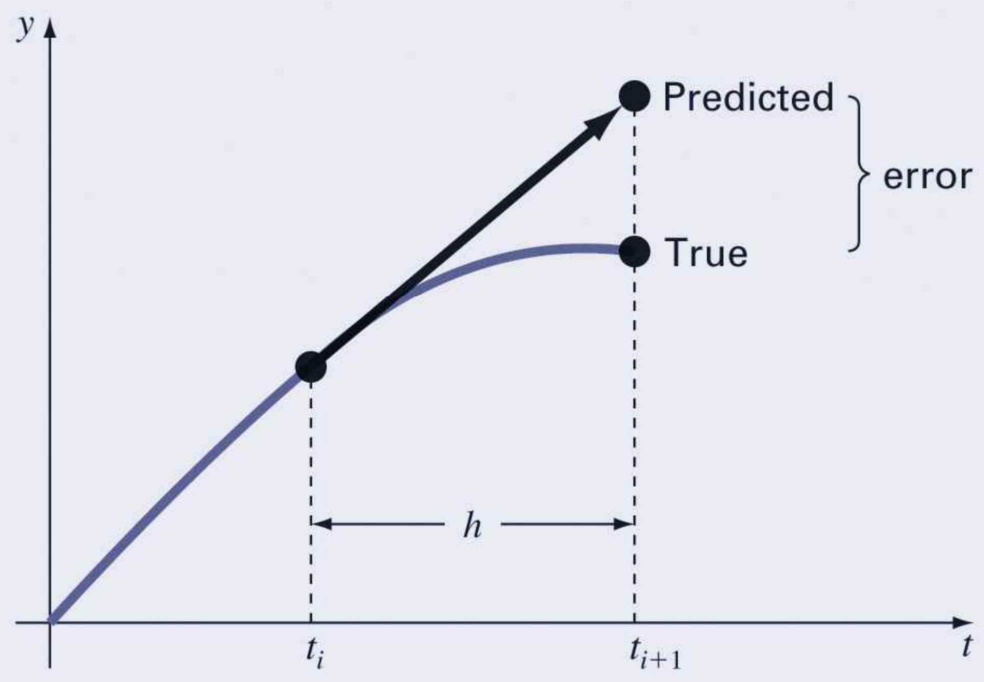
- The first derivative provides a direct estimate of the slope at t_i:

$$\left.\frac{dy}{dt}\right|_{t_i} = f(t_i, y_i)$$

- and the Euler method uses that estimate as the increment function:

$$\phi = f(t_i, y_i)$$

$$y_{i+1} = y_i + f(t_i, y_i)h$$



Example

Solve $y' = 4e^{0.8t} - 0.5y$ with $y(0)=2$.

$$\phi = f(t_i, y_i)$$

$$y_{i+1} = y_i + f(t_i, y_i)h$$

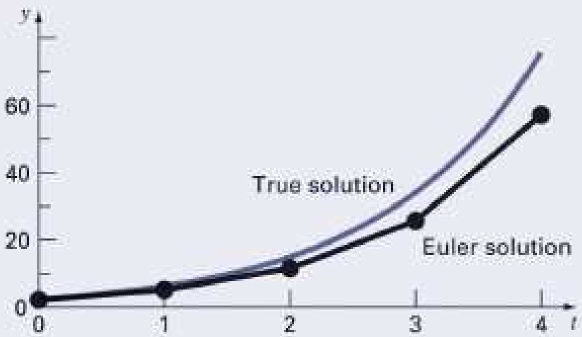
y(1) = y(0) + f(0,2)(1), where $f(0,2) = 4e^{0} - 0.5(2) = 3$

$\therefore\ y(1) = 2+3(1) = 5$ (exact value at $t=1$ is 6.19463)

y(2) = y(1) + f(1,5)(1), where $f(1,5) = 4e^{0.8(1)} - 0.5(5) = 6.4.216$

$\therefore\ y(1) = 5+ 6.4.216\ (1) = 11.40216$ (exact value at $t=2$ is 14.84392)

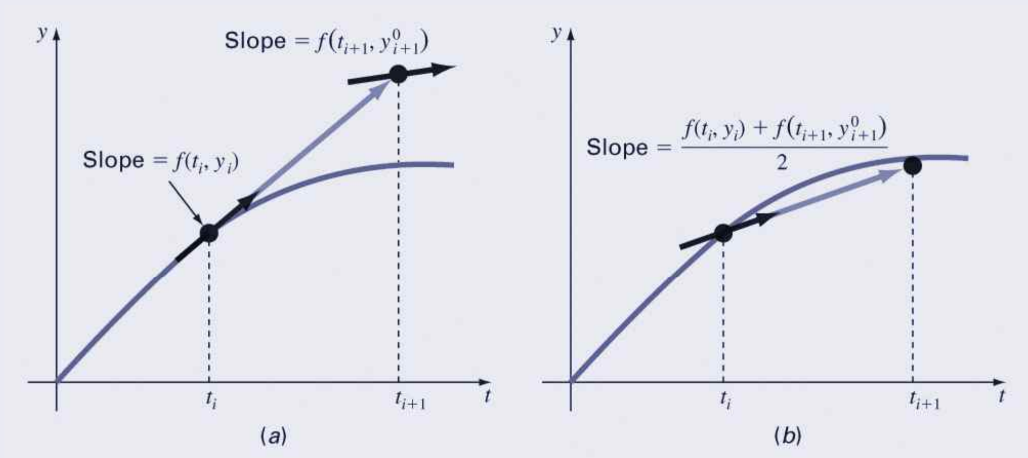
t	y_{true}	y_{Euler}	$\lvert\varepsilon_t\rvert$ (%)
0	2.00000	2.00000	
1	6.19463	5.00000	19.28
2	14.84392	11.40216	23.19
3	33.67717	25.51321	24.24
4	75.33896	56.84931	24.54



Analytical solution: $y = 4/1.3\ (e^{0.8t} - e^{-0.5t}) + 2e^{-0.5t}$

Heun's Method

- One method to improve Euler's method is to determine two derivatives at the beginning and predicted ending of the interval and average them:



- This process relies on making a prediction of the new value of y, then correcting it based on the slope calculated at that new value.
- This **predictor-corrector** approach can be iterated to convergence:

Heun's Method

기울기 추정값을 개선하기 위해 구간 초기점과 끝점에서 도함수를 구하여 평균을 취함

- **1**단계: 시작점 기울기 계산

$$y_i' = f(x, y)$$

- **2**단계: 시작점 기울기로 부터 끝점 예측

$$y_{i+1}^0 = y_i + f(x, y)h$$

- **3**단계: 예측 끝점 이용하여 끝점 기울기 계산

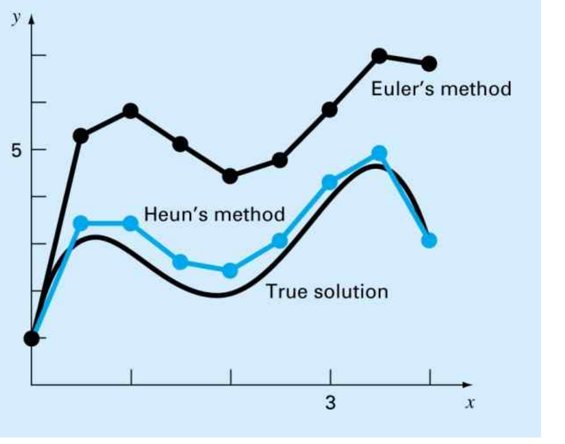
$$y_{i+1}' = f(x_{i+1}, y_{i+1}^0)$$

- **4**단계: 시작점과 끝점의 평균 기울기 계산

$$\overline{y'} = \frac{y_i' + y_{i+1}'}{2} = \frac{f(x_i, y_i) + f(x_{i+1}, y_{i+1}^0)}{2}$$

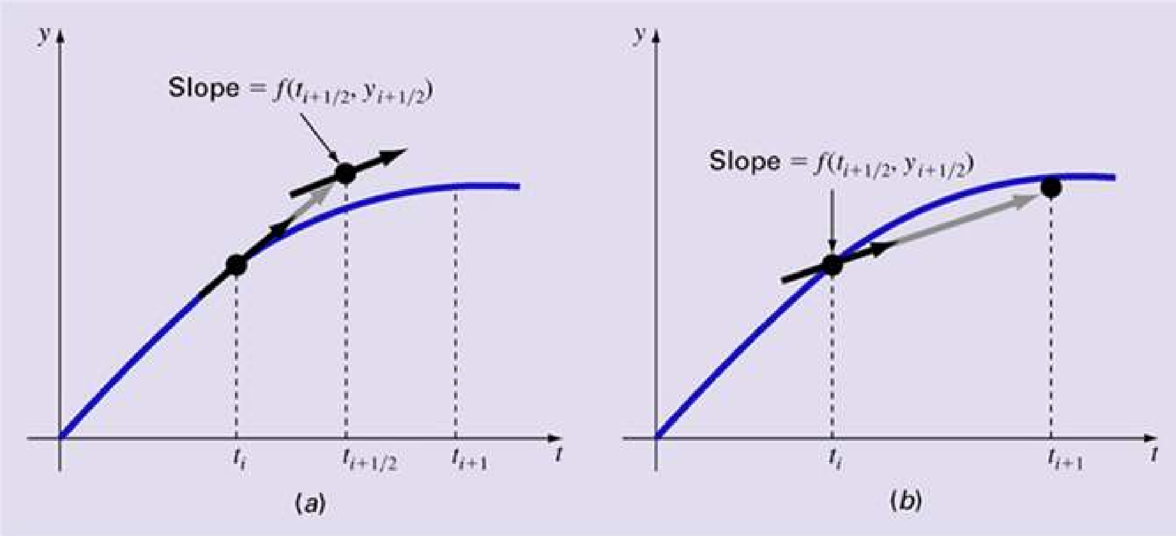
- **5**단계: 평균 기울기 이용하여 수정 끝점 예측

$$y_{i+1} = y_i + \frac{f(x_i, y_i) + f(x_{i+1}, y_{i+1}^0)}{2} h$$



Midpoint Method

- Another improvement to Euler's method is similar to Heun's method, but predicts the slope at the midpoint of an interval rather than at the end:



predictor : $y_{i+1/2} = y_i + f(t_i, y_i)\frac{h}{2}$

corrector : $y_{i+1} = y_i + f(t_{i+1/2}, y_{i+1/2})h$

- This method has a local truncation error of $O(h^3)$ and global error of $O(h^2)$

Runge-Kutta Methods

- Runge-Kutta (RK) methods achieve the accuracy of a Taylor series approach without requiring the calculation of higher derivatives.

- For RK methods, the increment function(slope) ϕ can be generally written as:

$$\phi = a_1 k_1 + a_2 k_2 + \cdots + a_n k_n$$

where the *a*'s, *p*'s and *q*'s are constants.

Runge-Kutta Methods

- General Form

$$y_{i+1} = y_i + \boxed{\phi(x_i, y_i, h)}h$$

increment function : 대표 기울기

Increment function

$$\phi = a_1k_1 + a_2k_2 + \cdots\cdots + a_nk_n \quad (a_i = \text{constants})$$

Recurrence relationships

$$k_1 = f(x_i, y_i)$$
$$k_2 = f(x_i + p_1h,\ y_i + q_{11}k_1h)$$
$$k_3 = f(x_i + p_2h,\ y_i + q_{21}k_1h + q_{22}k_2h) \quad (p, q = \text{constants})$$
$$\cdot$$
$$k_n = f(x_i + p_{n-1}h,\ y_i + q_{n-1,1}k_1h + q_{n-1,2}k_2h + \cdots + q_{n-1,n-1}k_{n-1}h)$$

2^{nd} order RK : Heun's and Midpoint methods

$$y_{i+1} = y_i + (a_1 k_1 + a_2 k_2)h$$

$$k_1 = f(x_i, y_i)$$
$$k_2 = f(x_i + p_1 h,\ y_i + q_{11} k_1 h)$$

derived by Chapra and Canale (2006)

$$\begin{cases} a_1 + a_2 = 1 \\ a_2 p_1 = \dfrac{1}{2} \\ a_2 q_{11} = \dfrac{1}{2} \end{cases}$$

3 equations 4 unknowns need to fix 1 unknown

$$\begin{cases} a_1 = 1 - a_2 \\ p_1 = q_{11} = \dfrac{1}{2a_2} \end{cases}$$

① Heun Method without Iteration $\left(a_2 = \frac{1}{2}\right)$

$$a_1 = \frac{1}{2}$$

$$p_1 = q_{11} = 1$$

$$y_{i+1} = y_i + \left(\frac{1}{2}k_1 + \frac{1}{2}k_2\right)h$$
$$k_1 = f(x_i, y_i)$$
$$k_2 = f(x_i + h, y_i + k_1 h)$$

② The Midpoint Method $(a_2 = 1)$

$$a_1 = 0$$

$$p_1 = q_{11} = \frac{1}{2}$$

$$y_{i+1} = y_i + k_2 h$$
$$k_1 = f(x_i, y_i)$$
$$k_2 = f\left(x_i + \frac{1}{2}h, y_i + \frac{1}{2}k_1 h\right)$$

2^{nd} order RK : Ralston's method

③ Ralston's method $\left(a_2 = \frac{2}{3}\right)$

$$a_1 = \frac{1}{3}$$

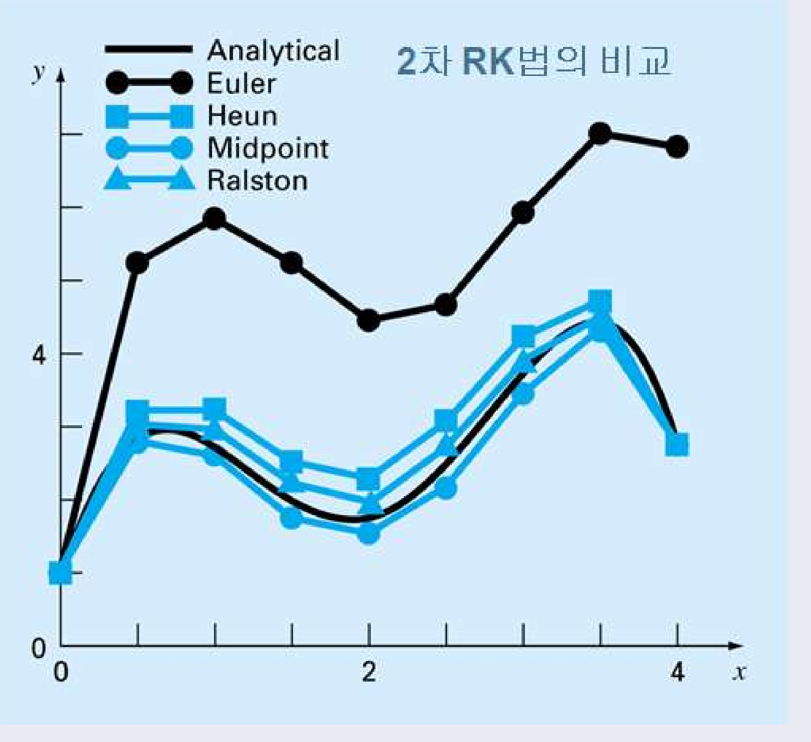
$$p_1 = q_{11} = \frac{3}{4}$$

$$y_{i+1} = y_i + \left(\frac{1}{3}k_1 + \frac{2}{3}k_2\right)h$$

$$k_1 = f(x_i, y_i)$$

$$k_2 = f\left(x_i + \frac{3}{4}h,\ y_i + \frac{3}{4}k_1 h\right)$$

$$\left[\begin{array}{l} a_1 = 1 - a_2 \\ p_1 = q_{11} = \dfrac{1}{2a_2} \end{array}\right.$$



3rd order RK

- 3rd RK method form

$$\boxed{y_{i+1} = y_i + \frac{1}{6}(k_1 + 4k_2 + k_3)h}$$

$$k_1 = f(x_i, y_i)$$

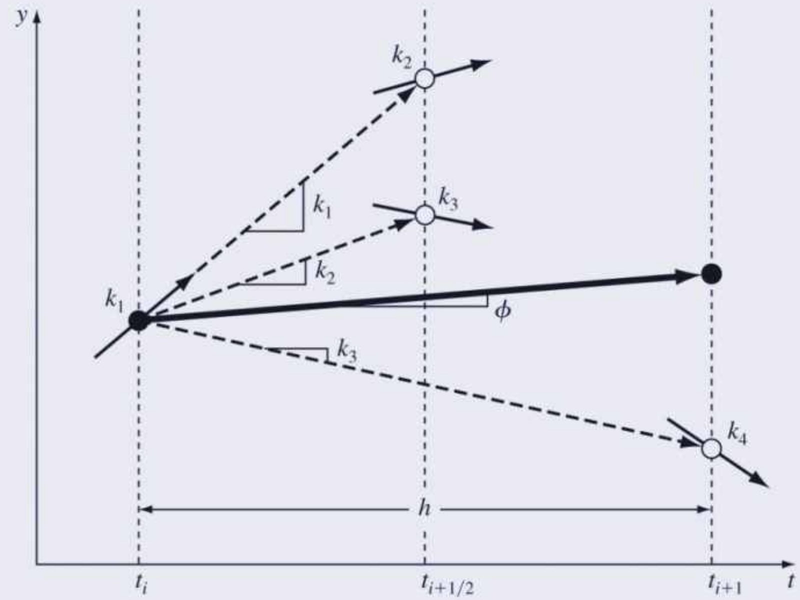
$$k_2 = f(x_i + \frac{1}{2}h,\ y_i + \frac{1}{2}k_1 h)$$

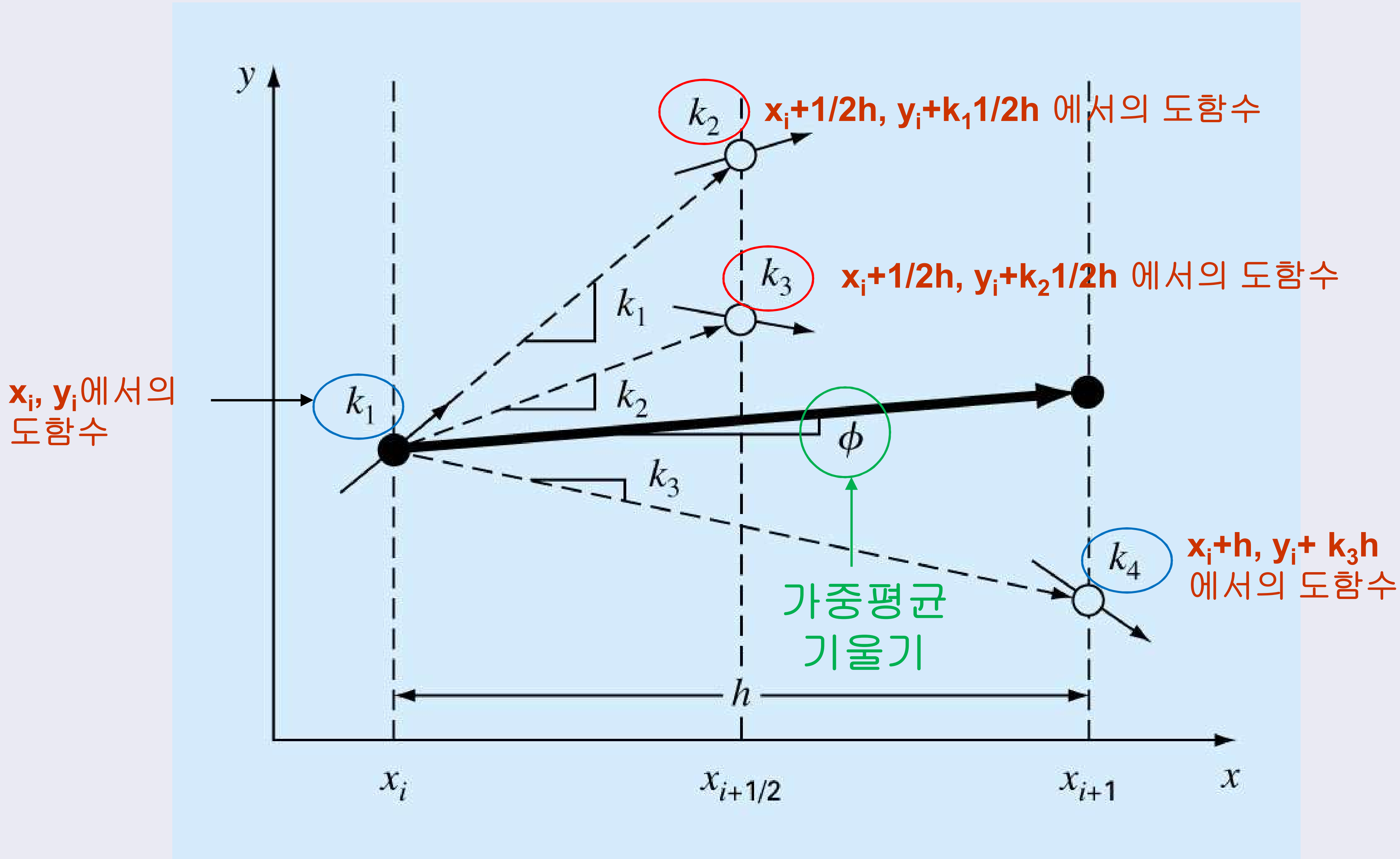
$$k_3 = f(x_i + h,\ y_i - k_1 h + 2k_2 h)$$

Classical Fourth-Order Runge-Kutta Method

- The most popular RK methods are fourth-order
  - The most commonly used form is:

$$y_{i+1} = y_i + \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)h$$





y
k2
xi+1/2h, yi+k11/2h 에서의 도함수
k3
xi+1/2h, yi+k21/2h 에서의 도함수
k1
k2
xi, yi에서의 도함수
k1
φ
k3
k4
xi+h, yi+ k3h 에서의 도함수
가중평균 기울기
h
xi
xi+1/2
xi+1
x

MATLAB Functions

- MATLAB's ode23 function uses second- and third-order RK functions to solve the ODE and adjust step sizes. (Bogacki and Shampine algorithm)
  - `[t,y]= ode23(odefun,[ti tf], y0)`
    - [t,y] : solution array, where each column represents one of the variables and each row corresponds to a time in the t vector
    - odefun : function returning a column vector of the right-hand-sides of the ODEs
    - tspan : time over which to solve the system

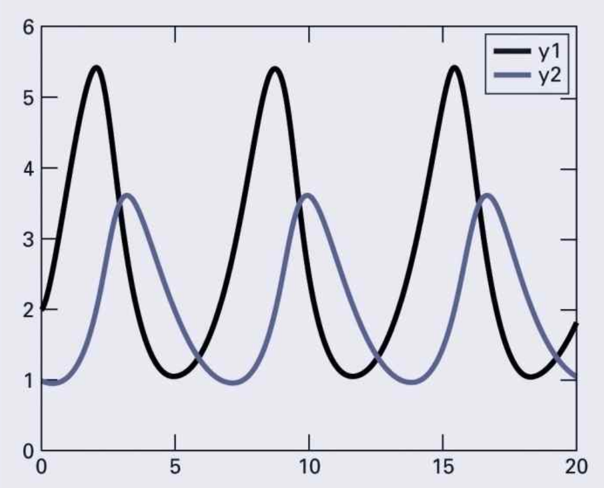
- MATLAB's ode45 function uses fourth- and fifth-order RK functions to solve the ODE and adjust step sizes. (Dormand and Prince algorithm)
  - This is recommended as the first function to use to solve a problem.
  - `ode45(odefun,[ti tf], y0)`

- MATLAB's ode113 function is a multistep solver useful for computationally intensive ODE functions. (Variable order Adams-Bashforth-Moulton algorithm)
  - `ode113(odefun,[ti tf], y0)`

Example - Predator-Prey

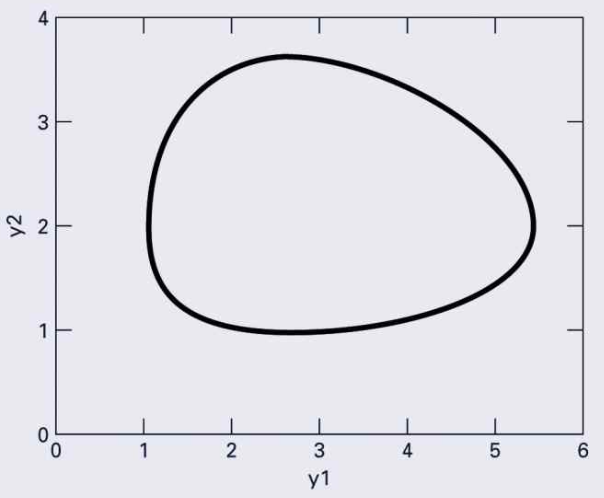
- Solve: $\frac{dy_1}{dt} = 1.2y_1 - 0.6y_1y_2 \quad \frac{dy_2}{dt} = 0.8y_2 + 0.3y_1y_2$

  with $y_1(0)$=2 and $y_2(0)$=1 for 20 seconds



- predprey.m M-file

```
function yp = predprey(t, y)
yp = [1.2*y(1)-0.6*y(1)*y(2);...
     -0.8*y(2)+0.3*y(1)*y(2)];
tspan = [0 20];
y0 = [2, 1];
[t, y] = ode45(@predprey, tspan, y0);
figure(1); plot(t,y);
figure(2); plot(y(:,1),y(:,2));
```



Part 6

Ordinary Differential Equations

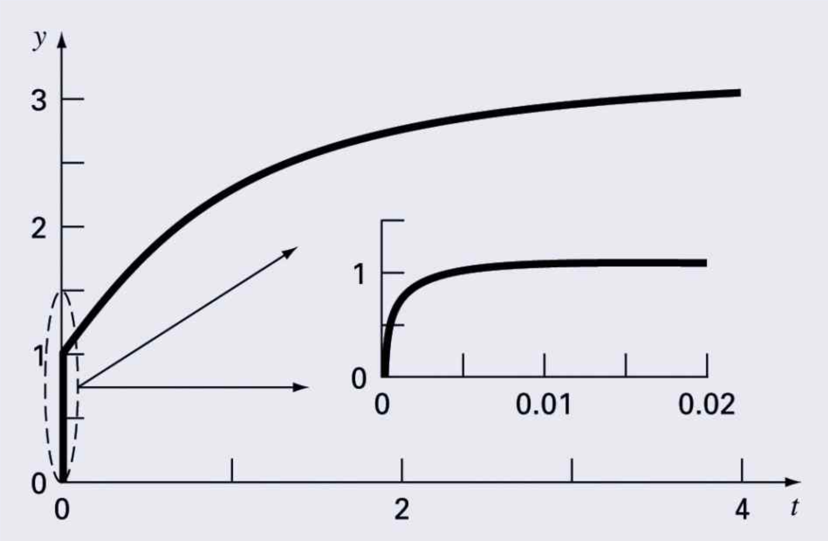
Chapter 23 : Adaptive Methods and Stiff Systems

Stiffness

- A *stiff system* is one involving rapidly changing components together with slowly changing ones.

- An example of a single stiff ODE is: $\dfrac{dy}{dt} = -1000y + 3000 - 2000e^{-t}$

  whose solution if *y(0)=0* is: $y = 3 - 0.998e^{-1000t} - 2.002e^{-t}$



Implicit Euler's Method

$$\frac{dy}{dt} = -ay$$
$$y(0) = y_0$$

$$y = y_0 e^{-at}$$

Explicit method

$$y_{i+1} = y_i + \frac{dy_i}{dt} h$$
$$= y_i - a y_i h$$
$$= y_i (1 - ah)$$

$|1 - ah| < 1$ *stable*

Implicit method

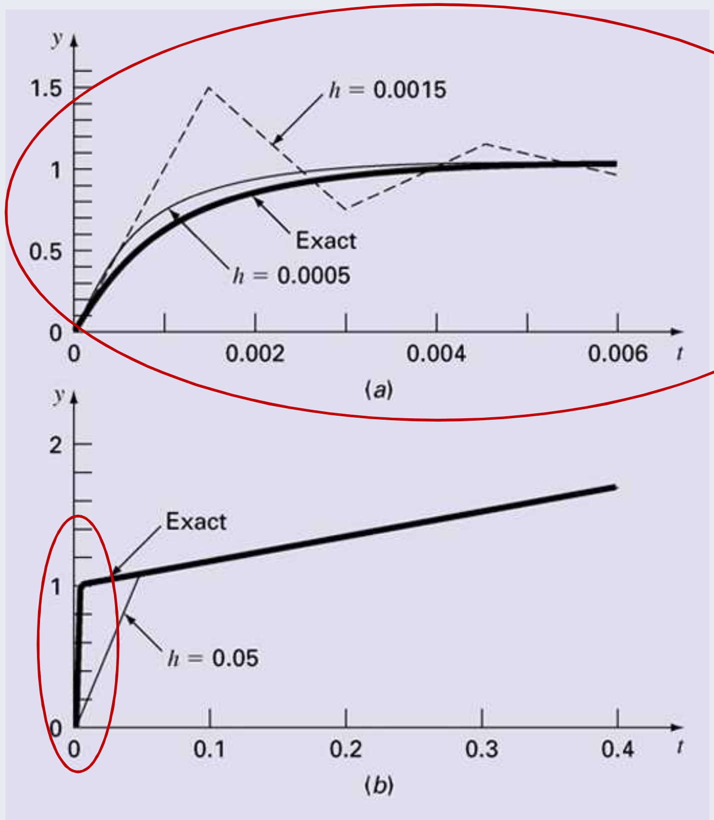
$$y_{i+1} = y_i + \frac{dy_{i+1}}{dt} h$$
$$= y_i - a y_{i+1} h$$
$$\therefore y_{i+1} = y_i \left(\frac{1}{1 + ah} \right)$$

unconditionally stable

Explicit vs. Implicit Euler's Method

$$\frac{dy}{dt} = -1000y + 3000 - 2000e^{-t}, y(0) = 0$$

$$y = 3 - 0.998e^{-1000t} - 2.002e^{-t}$$



explicit

implicit

MATLAB Functions for Stiff Systems

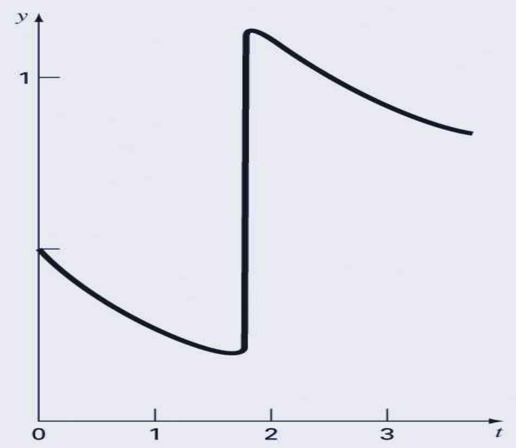
- MATLAB functions for solving stiff systems of ODEs
  - ode15s, ode23s, ode23t, ode23tb

- `ode23tb(odefun,[ti tf], y0);`
  - implicit RK methods
  - [ti tf] : time interval
  - y0 : initial condition vector

```
% The lorenz function is defined by:
% function dy = lorenz(times, y)
% dy = zeros(3, 1);
% dy(1) = 10*(y(2)-y(1));
% dy(2) = 28*y(1)-y(2)-y(1)*y(3);
% dy(3) = y(1)*y(2)-8/3*y(3);
[t, y] = ode23tb('lorenz', [0, 5], [1; 1; 1])
```

Adaptive Runge-Kutta Methods

- The solutions to some ODE problems exhibit multiple time scales
  - for some parts of the solution the variable changes slowly, while for others there are abrupt changes.
- Constant step-size algorithms would have to apply a small step-size to the entire computation
  - wasting many more calculations on regions of gradual change.
- Adaptive algorithms, on the other hand, can change step-size depending on the region.



Approaches to Adaptive Methods

- There are two primary approaches to incorporate adaptive step-size control:
  - *Step halving* : perform the one-step algorithm two different ways, once with a full step and once with two half-steps, and compare the results.
    - Error estimate : difference between estimates based on one full step and two half steps
  - *Embedded RK methods* (Fehlberg RK methods) : perform two RK iterations of different orders and compare the results. (this is the preferred method)
    - Error estimate : difference between two predictions based on different order RK methods

Part 6

Ordinary Differential Equations

Chapter 24

Boundary-Value Problems

Concept

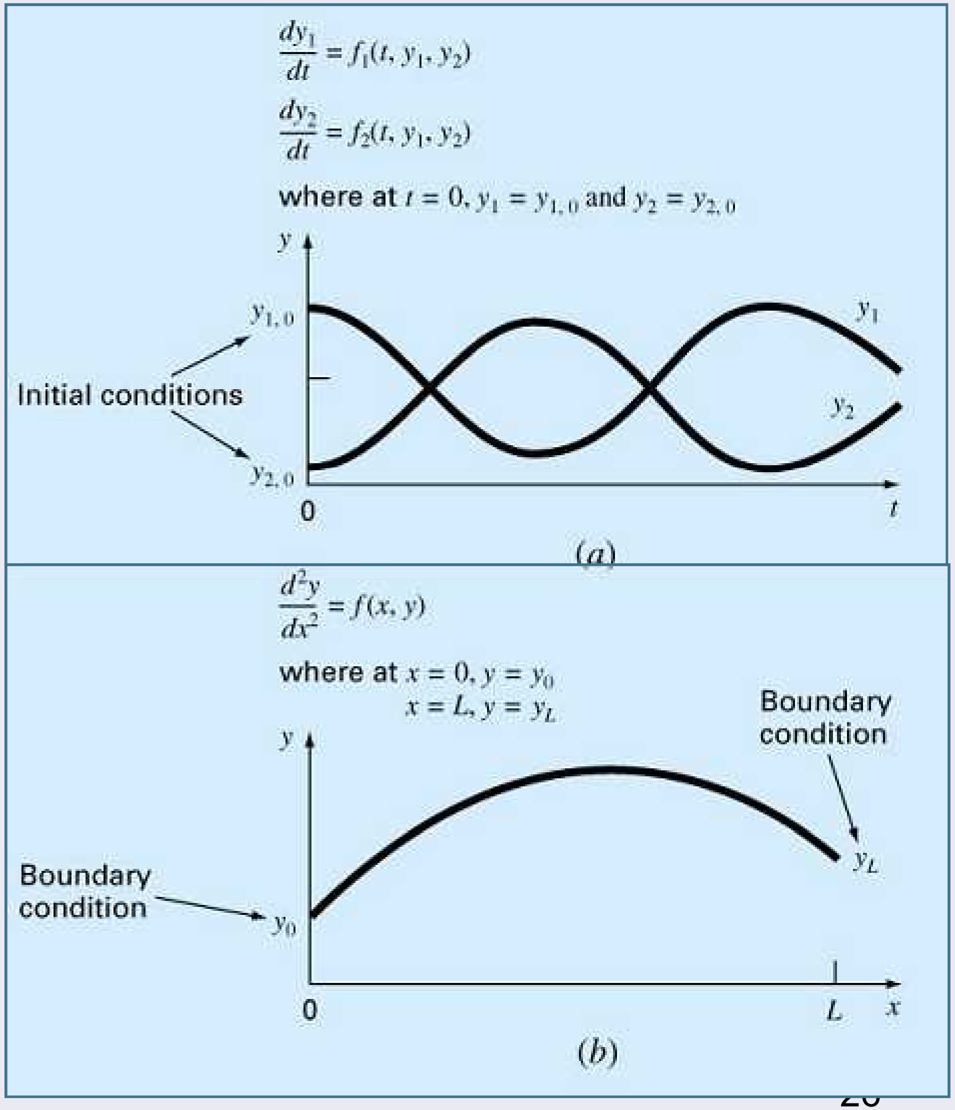
❑ 초기값 문제(IVP)

모든 조건이
독립변수의
일정한 값에서
지정

❑ 경계값 문제(BVP)

모든 조건이
독립변수의 다른
값에서 지정

(일반적으로 끝단/경계
조건 지정)

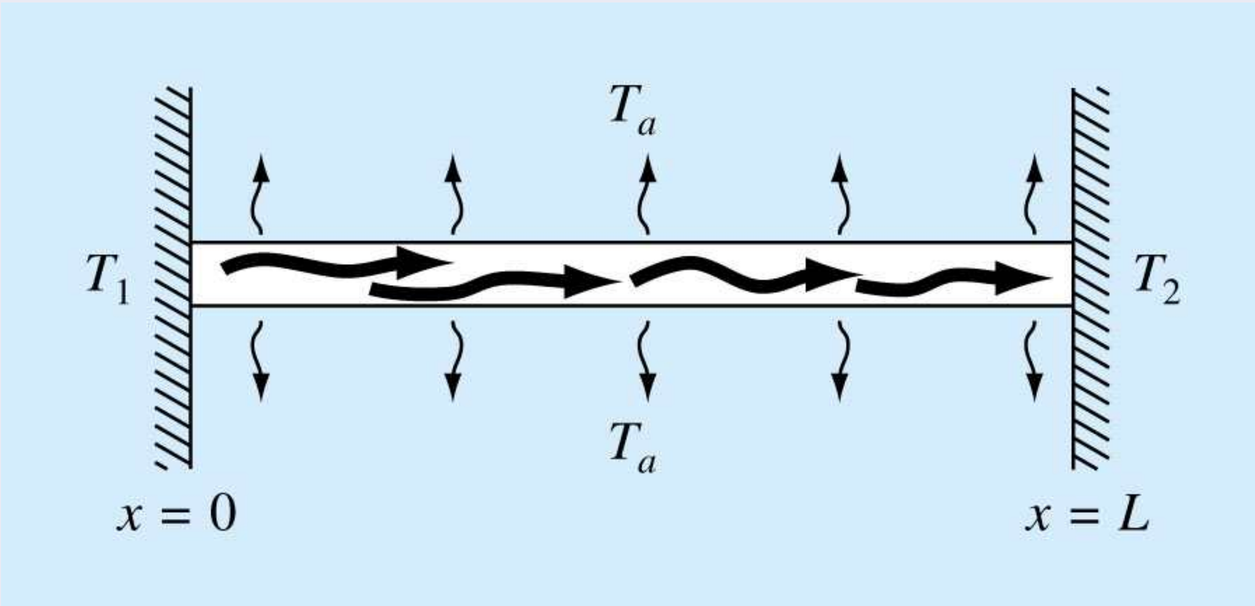


Example of BVP

- **Heat Transfer**

$$\frac{d^2T}{dx^2} + h^{'}(T_a - T) = 0 \qquad h^{'} : 열전달율 \; T_a : 공기온도$$

경계조건 $T(x=0) = T_1, \quad T(x=L) = T_2$



❑ Shooting Method

- 경계값문제를 초기값문제로 변환하여 시행착오법 적용

$$\frac{d^2T}{dx^2} + h'(T_a - T) = 0 \qquad h' = 0.01m^{-2},\ T_a = 20\,^oC,\ L = 10m$$

경계조건 T(x =0) = 40, T(x=10) = 200

1단계**: 2**차 방정식을 **2**개의 **1**차 상미분 방정식으로 변환

$$\frac{dT}{dx} = z, \qquad \frac{dz}{dx} = h'(T - T_a)$$

온도변화율

2단계**: 4**차 **RK**법 등으로 초기값문제를 시행착오법 적용

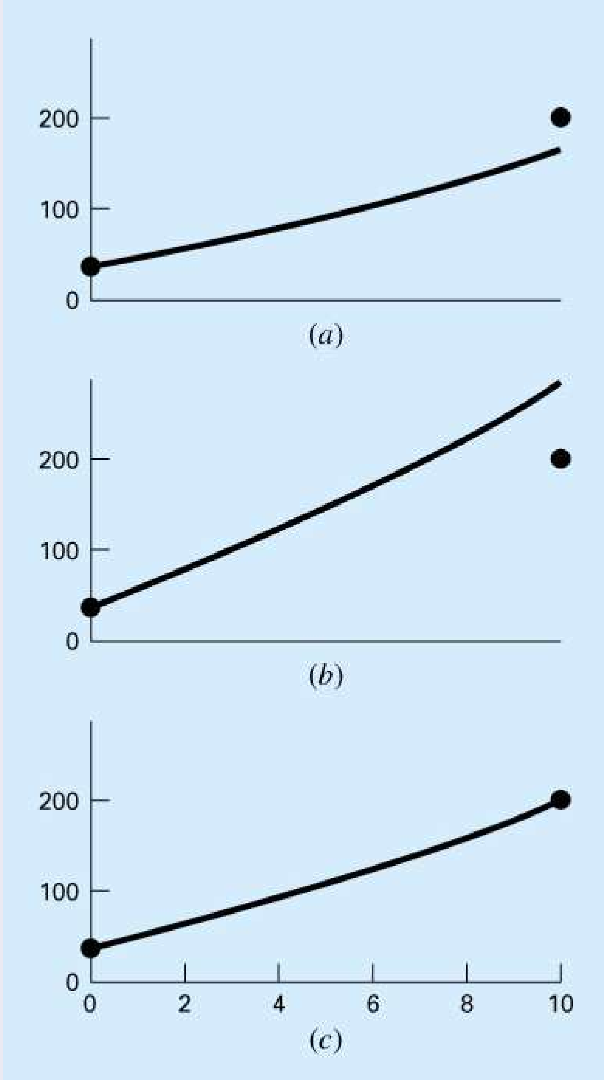
❑ Shooting method : 명중할 때까지 z(0) 가정

z(0) = 10 가정, **T(x=10) = 168.3797**

z(0) = 20 가정, **T(x=10) = 285.8980**

- 선형보간으로 **T(10)=200** 만족하는 **z(0)** 구함

$$z(0) = 10 + \frac{20 - 10}{285.8980 - 168.3797}(200 - 168.3797) = 12.6907$$



Finally…
The Semester Ends…

익숙치 않은 공부하느라
모두 고생 많았습니다…
앞으로 여러분에게
결정적인 기회에
큰 보상이 있을 것입니다…
Good Luck…