Euler Method for (IVP) $\begin{cases} y' = f(t,y) & , t_0 < t < t_{end} \\ y(t_0) = y_0 \end{cases}$

Idea: Follow the direction field (slopes)

$$y'(t) \approx \frac{y(t+\Delta t) - y(t)}{\Delta t}$$, the smaller Δt the better the approximation

$$\Rightarrow \quad y(t+\Delta t) \approx y(t) + \Delta t \cdot f(t, y(t))$$

Discretization: discretize time interval into discrete time steps

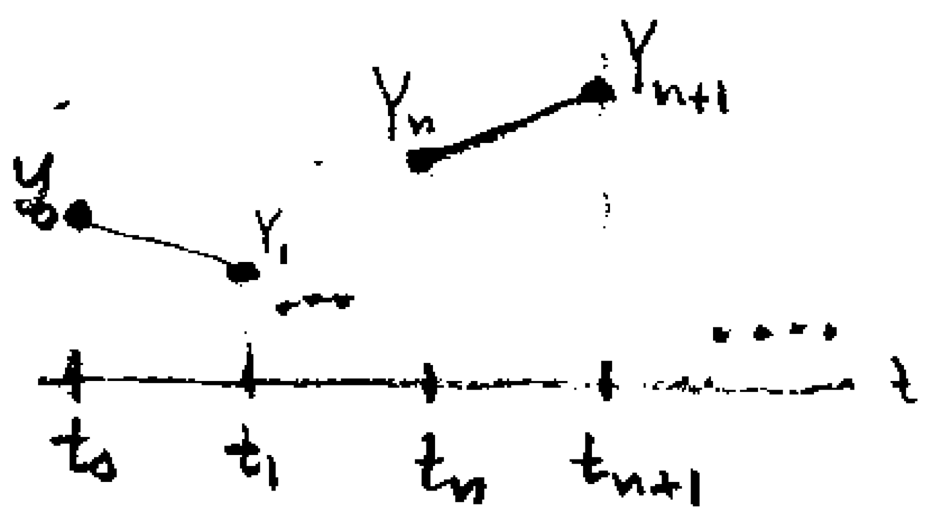
$$t_0 < t_1 < t_2 < \dots < t_N = t_{end} \ , \quad \Delta t_n = t_{n+1} - t_n$$

Construct values $Y_n \approx y(t_n)$, $n = 0:N$ by a numerical method.

Euler Method: $Y_0 = y_0$ (given by IC)

$$Y_{n+1} = Y_n + \Delta t_n \cdot f(t_n, Y_n) \ , \ n = 0, 1, \dots, N-1$$

It is a timestepping scheme: knowing Y_n it produces Y_{n+1}



Forward Euler scheme, the simplest method for numerical approximation of the solution to a Well-posed IVP.

Explicit scheme: RHS contains known t_n, Y_n, only need to evaluate it.

Backward Euler: $Y_{n+1} = Y_n + \Delta t \cdot f(t_{n+1}, Y_{n+1})$ must solve equation for Y_{n+1}

implicit scheme: must solve for Y_{n+1}.

Coding

1. $f(t,y)$ should be coded in a function subprogram to return value of $f(t_n, Y_n)$

```
function  Fn = FCN ( tn, Yn)
              Fn = (formula for f(tn, Yn) )
end
```

Euler scheme

2. read input data : N_{max} , t_0 , y_0 , t_{end}

3. Set $\Delta t = \dfrac{t_{end} - t_0}{N_{max}}$

4. Initialize : $tn = t_0$, $Yn = y_0$, ERRmax = 0.0

5. Time stepping:

```
for n = 0 : Nmax

    Yn = Yn + Δt · FCN ( tn, Yn)
    tn = tn + Δt          (better: tn = to + (n+1)·Δt )

    print: tn , Yn     (use fprintf , formatted %f %f ..
  or, during debugging knowing exact solution:
  | COMPARE ( tn , Yn , )
          Yexact = ... formula (tn)
          ERRn = abs( Yn − Yexact)
          ERRmax = max( ERRmax , ERRn)

          print: tn Yn Yexact ERRn

end for

print: 'Done.: t =', tn , 'ERRmax =', ERRmax
```