- **Euler's Method**
  - Numerical method that uses linear approximation with steps to approximate the behavior of the solution to the differential equation.
  - Requires a differential equation and a point (i.e. initial value problem).
  - Uses very small steps for *dx*. $dy = f(x, y)dx$
  - Algorithm:
    - Start at $x = x_o$, $y = y_o$. Step size *dx*.
    - Calculate $f(x, y)$.
    - Calculate $dy = f(x, y)dx$.
    - $x_{n+1} = x_n + dx$, $y_{n+1} = y_n + dy$
    - Repeat.
  - Use a table! Have a column for *x*, *y*, $f(x, y)$, and *dy*.
  - The smaller the steps, the more accurate the numerical approximation (less error).
  - Relation of error to concavity
    - Does not take into account concavity as it uses linear approximation
    - Concave up = underestimate
    - Concave down = overestimate
    - More concavity = more error
  - Algorithm runs in $O(n)$. i.e. double the steps, half the error
- **Heun's Method or Improved (Modified) Euler's Method or Runge-Kutta 2 (RK2)**
  - Similar to Euler's Method, but uses better slope by averaging the two slopes at the beginning of *dx* and at the end of *dx*.
  - Algorithm:
    - Start at $x = x_o$, $y = y_o$. Step size *dx*.
    - Calculate $m_1 = f(x_n, y_n)$. This is slope 1.
    - Calculate $dy = f(x, y)dx$.
    - $\tilde{y}_{n+1} = y_n + dy$. $\tilde{y}$ is the guess of the next *y* using Euler's method.
    - Calculate $m_2 = f(x_{n+1}, \tilde{y}_{n+1})dx$. This is slope 2.
    - $dy = \left(\frac{m_1 + m_2}{2}\right)dx$. Take the average of the two slopes
    - $x_{n+1} = x_n + dx$, $y_{n+1} = y_n + dy$
    - Repeat.
  - Algorithm runs in $O(n^2)$. i.e. double the steps, quarter the error

Further Notes:

- There are Runge-Kutta algorithms of higher orders
  - For example, RK4 would calculate 4 slopes and find their weighted average to calculate the next *y*.
  - RK4 is the standard algorithm for most computer programs (e.g. MATLAB) to numerically solve first-order differential equations