EngMath - ps_1

A. Q. Snyder

October 29, 2023

1 HW template

For more details, the entire template and its supplementary materials can be found on GitHub: here. Under the ps_1 directory exist the necessary files for this LaTeX generation, the compiled plots, MATLAB exports, Python exports, the virtual Jupyter kernels used to prototype Python code, and all MATLAB files.

width=!,height=!,pages=-

2 Hand Calcs

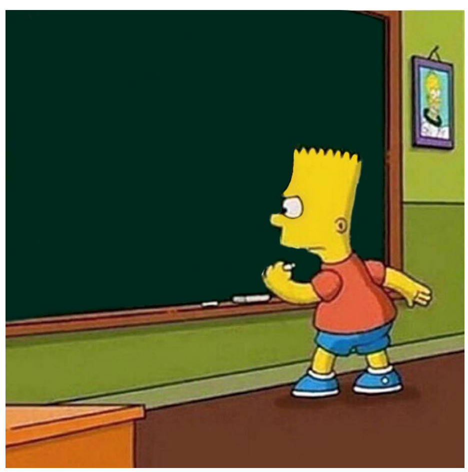


Figure 1: Hand Calculations

1.

$$\boxed{m\left(\frac{dv}{dt}\right) = mg - c(v)^2 \ ; \quad v(0) = 0} \sim \frac{dv}{dt} = 0 \ @ \ V_s$$

$$\frac{dv}{dt} = g - \frac{c(v)^2}{m}$$

Variables: v, t
parameters: g, c, m

$$\tilde{v} = \frac{v - \cancel{v_r}}{v_s} \qquad \tilde{t} = \frac{t - \cancel{t_r}}{t_s}$$

$\_{r}$ = reference value
$\_{s}$ = scaling factor

$v(0) = 0 \rightarrow v_r, t_r = 0$

$$\frac{dv}{dt} = \frac{d(\tilde{v} \cdot v_s)}{d(\tilde{t} \cdot t_s)} \longrightarrow \frac{v_s}{t_s} \frac{d\tilde{v}}{d\tilde{t}}$$

$$m \frac{v_s}{t_s} \frac{d\tilde{v}}{d\tilde{t}} = mg - c v_s^2 \tilde{v}^2$$

non-dim

$$\left[m \frac{v_s}{t_s} = mg \qquad t_s = \frac{v_s}{g} \right]$$

$$m \frac{v_s}{\frac{v_s}{g}} \frac{d\tilde{v}}{d\tilde{t}} = mg - c v_s^2 \tilde{v}^2$$

— divide by mg

$$\frac{d\tilde{v}}{d\tilde{t}} = 1 - \frac{c}{mg} v_s^2 \tilde{v}^2$$

— choose v_c to be terminal velocity

$$\frac{dv}{dt} = 0 = g - \frac{cv^2}{m}$$

$$v_c^2 = \frac{mg}{c}$$

$$\frac{d\tilde{v}}{d\tilde{t}} = 1 - \tilde{v}^2$$

- plot with Runge-kutta
- set $v = 0.95$
- check $v = 1 \rightarrow \frac{dv}{dt}$ should $= 0$

2. $y'' + 9y = 0$ $\quad y(5) + 14y'(5) = 10 \quad y(1) = 0$ $\quad K = 9$, $I = 14$

$$y = \sum_{n=0}^{\infty} a_n x^n \quad y' = \sum_{n=1}^{\infty} n a_n x^{n-1} \quad y'' = \sum_{n=2}^{\infty} n(n-1) a_n x^{n-2}$$

$$\sum_{n=2}^{\infty} n(n-1) a_n x^{n-2} + 9 \sum_{n=0}^{\infty} a_n x^n = 0$$

$p = n-2$, $n = p+2$ $\qquad$ $p = n$, $n = p$

$$\sum_{p=0}^{\infty} (p+2)(p+1) a_{p+2} x^p + 9 \sum_{p=0}^{\infty} a_p x^p = 0$$

$$\sum_{p=0}^{\infty} \left[(p+2)(p+1) a_{p+2} + 9 a_p \right] x^p = 0 \implies p = 0, 1, 2, 3 \ldots$$

$$a_{p+2} = \frac{-9 a_p}{(p+2)(p+1)}$$ recurrence

code
$\left(\begin{array}{l} idx = p+2 \\ p = idx - 2 \end{array} \right)$

$a_2 = \dfrac{-9a_0}{2} = 0 \quad p = 0$

$a_3 = \dfrac{-9a_1}{6} = \dfrac{-9a_1}{6} \quad p = 1$

$a_4 = \dfrac{-9a_2}{12} = 0 \quad p = 2$

$a_5 = \dfrac{-9a_3}{20} \quad p = 3$

$a_6 = 0 \quad p = 4$

$a_7 = \dfrac{-9a_5}{42} \quad p = 5$

3. $x^2y'' + xy' - y = 9x^2 + 41x \qquad y(1)=0 \quad y(5)=14$

find two linearly independent soln. $(y_1 \& y_2)$ for the homogenous part

$$x^2y'' + xy' - y = 0$$

Cauchy-Euler equation!!

$\hookrightarrow$ soln. $y(x) = x^m$

$$\Downarrow$$

$$m(m-1) + m - 1 = 0$$

roots: $m_1 = 1 \quad m_2 = -1 \longrightarrow$ linearly independent solutions $\begin{bmatrix} y_1(x) = x \\ y_2(x) = 1/x \end{bmatrix}$

Wronskian: $W(y_1, y_2) = \begin{vmatrix} y_1 & y_2 \\ y_1' & y_2' \end{vmatrix}$

$$W(y_1, y_2) = \begin{vmatrix} x & 1/x \\ 1 & -1/x^2 \end{vmatrix} = -2$$

particular soln: $y_p(x) = y_1(x)\int \frac{y_2(x)(kx^2+4x)}{W(y_1,y_2)}dx - y_2(x)\int \frac{y_1(x)(kx^2+4x)}{W(y_1,y_2)}dx$

expanding:

$$u_1(x) = \int \frac{1/x\,(kx^2+4x)}{-2}\,dx$$

$$u_2(x) = -\int \frac{x(kx^2+4x)}{-2}\,dx$$

$$y_p(x) = x \cdot u_1(x) + \frac{1}{x} \cdot u_2(x)$$

complete solution with Simpson integration

4. $xy'' + 2y' + y = 0 \qquad y(1) = -10 \qquad y(5) = -18$

- Frobenius: $y = x^{\lambda}\sum_{j=0}^{\infty} c_j x^j = \sum_{j=0}^{\infty} (c_j) x^{(\lambda+j)}$

$$\left[\begin{aligned} y' &= \sum_{j=0}^{\infty} (\lambda+j)(c_j)\, x^{\lambda+j-1} \\ y'' &= \sum_{j=0}^{\infty} (c_j)(\lambda+j)(\lambda+j-1) x^{\lambda+j-2} \end{aligned}\right]$$ (first and second deriv.)

- Solution: (by substitution into orig. ODE)

$$(x)\left[\sum_{j=0}^{\infty}(c_j)(\lambda+j)(\lambda+j-1)x^{\lambda+j-2}\right] + (2)\left[\sum_{j=0}^{\infty}(\lambda+j)(c_j)\,x^{\lambda+j-1}\right] + \left[\sum_{j=0}^{\infty}(c_j)x^{\lambda+j}\right] = 0$$

$$\cancel{x^{\lambda}}\left[\sum_{j=0}^{\infty}(c_j)(\lambda+j)(\lambda+j-1)x^{j-1} + \sum_{j=0}^{\infty}(\lambda+j)(c_j)2x^{j-1} + \sum_{j=0}^{\infty}(c_j)x^{j}\right] = 0$$

(get powers to match) $k=j \qquad k=j \qquad k=j+1$

$$\sum_{k=0}^{\infty}(c_k)(\lambda+k)(\lambda+k-1)x^{k-1} + \sum_{k=0}^{\infty}(\lambda+k)(c_k)2x^{k-1} + \sum_{k=1}^{\infty}(c_{k-1})x^{k-1}$$

$$c_k \sum_{k=0}^{\infty}(\lambda+k)(\lambda+k-1)x^{k-1} + (\lambda+k)2x^{k-1} + \sum_{k=1}^{\infty}(c_{k-1})x^{k-1}$$

(evaluate first two groups at $k=0$ to get matching indices

$$c_0\Big[\underbrace{\lambda(\lambda-1)x^{-1} + 2\lambda x^{-1}}_{=0}\Big] + \sum_{k=1}^{\infty}\Big[c_k(\lambda+k)(\lambda+k-1)+(\lambda+k)(2)+(c_{k-1})\Big]x^{k-1} = 0$$

indicial eqn for λ — recursion relation for c_j

$$\lambda^2+\lambda = 0 \longrightarrow \frac{-b\pm\sqrt{b^2-4ac}}{2a} \qquad \lambda_1 = 0 \qquad \lambda_2 = -1$$

given $y = x^{\lambda}\left[\sum_{j=0}^{\infty} c_j x^j\right]$

$$c_k = \frac{-c_{k-1}}{(\lambda+k)(\lambda+k-1)(2\lambda+2k}$$

$$y_1 = x^0\left[c_0 + c_1 x + c_2 x^2 + \dots\right]$$

$$y_2 = x^{-1}\left[d_0 + d_1 x + d_2 x^2 + \dots\right]$$

$$c_k = \frac{-c_{k-1}}{(k)(k-1)(2k)}$$

$$c_k = \frac{-c_{k-1}}{(k-1)(k-2)(2k-2)}$$

3 Python Outputs

Figure 2: Python Logo

1. **[10 pt]** If a mass is dropped from a very large height the governing equation of motion for the (downward) velocity V is:

$$\boxed{m\ \frac{dV}{dt} = m\ g - c\,(V)^2\,; \quad V\,(0) = 0}$$

Here, c is the coefficient of drag in air.

(a) Put this governing equation in dimensionless form. (Hint, think about the terminal velocity V_T.)
(b) Solve the problem in its dimensionless form.
(c) Determine when the mass achieves 95% of its terminal velocity

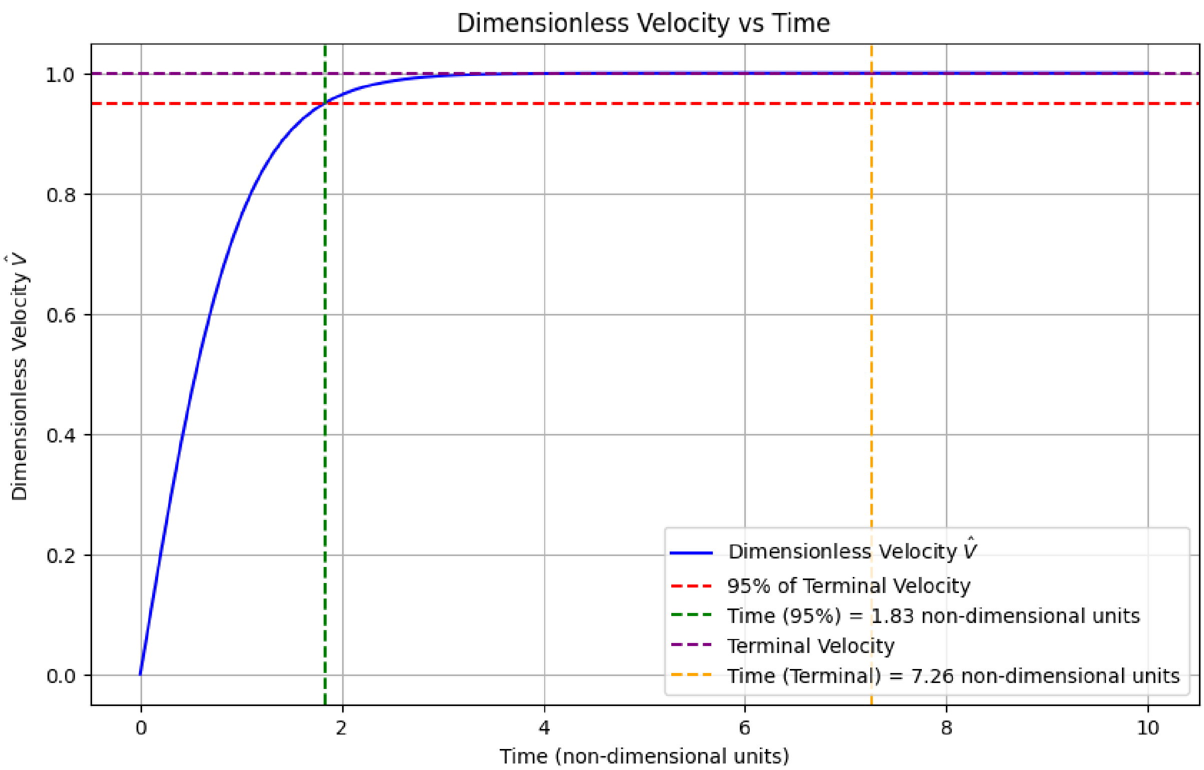
In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [ ]:
```
def model(v):
    dvdt = 1 - v**2
    return dvdt

def rk4(v, t, dt):
    k1 = dt * model(v)
    k2 = dt * model(v + 0.5 * k1)
    k3 = dt * model(v + 0.5 * k2)
    k4 = dt * model(v + k3)

    v_new = v + (k1 + 2*k2 + 2*k3 + k4) / 6.0
    return v_new

t = np.linspace(0, 10, 5000)
dt = t[1] - t[0]

v = np.zeros_like(t)
v[0] = 0

for i in range(1, len(t)):
    v[i] = rk4(v[i-1], t[i-1], dt)

# Handle for v=0.95
indices_95 = np.where(v >= 0.95)[0]
if indices_95.size > 0:
    idx_95 = indices_95[0]
    time_95 = t[idx_95]
    print(f"Velocity reaches 95% at {time_95:.2f} non-dimensional units.")
else:
    idx_95 = None
    time_95 = None

# Handle for v=1 (terminal velocity)
indices_1 = np.where(v >= 0.999999)[0]
if indices_1.size > 0:
    idx_1 = indices_1[0]
    time_1 = t[idx_1]
    print(f"Velocity reaches terminal velocity at {time_1:.2f} non-dimensional unit
else:
```

```
    idx_1 = None
    time_1 = None
```

```
Velocity reaches 95% at 1.83 non-dimensional units.
Velocity reaches terminal velocity at 7.26 non-dimensional units.
```

In [ ]:
```
plt.figure(figsize=(10, 6))
plt.plot(t, v, label=r"Dimensionless Velocity $\hat{V}$", color='blue')
if time_95:
    plt.axhline(0.95, color='red', linestyle='--', label="95% of Terminal Velocity"
    plt.axvline(time_95, color='green', linestyle='--', label=f"Time (95%) = {time_
if time_1:
    plt.axhline(1.0, color='purple', linestyle='--', label="Terminal Velocity")
    plt.axvline(time_1, color='orange', linestyle='--', label=f"Time (Terminal) = {
plt.xlabel('Time (non-dimensional units)')
plt.ylabel('Dimensionless Velocity $\hat{V}$')
plt.title('Dimensionless Velocity vs Time')
plt.legend()
plt.grid(True)
plt.show()
```



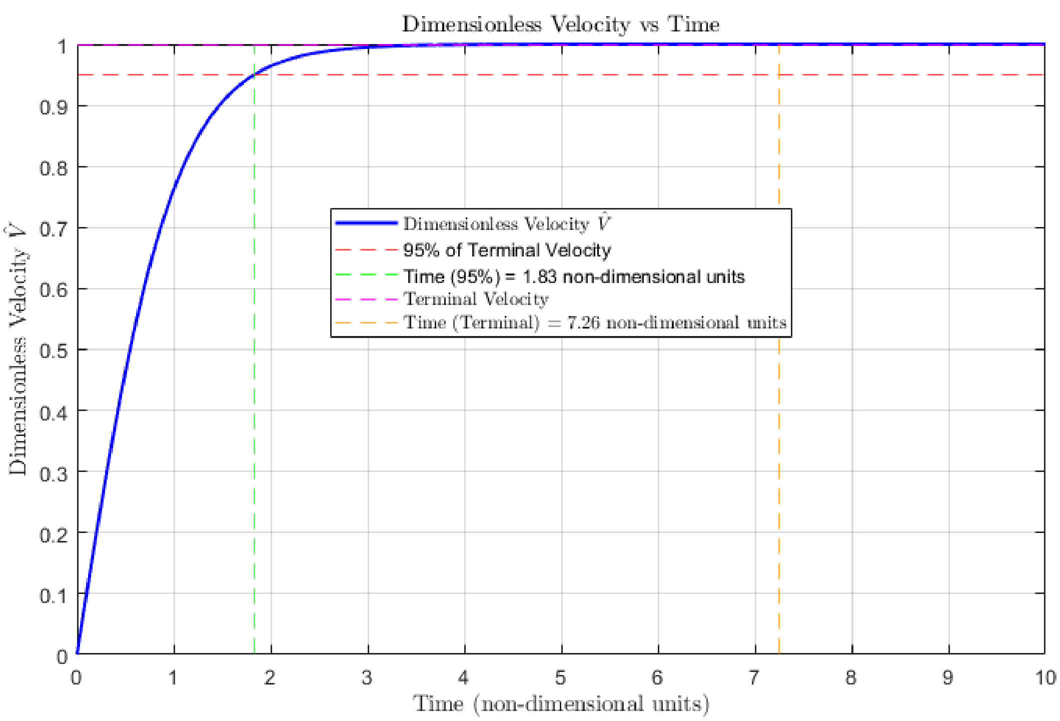
4 MATLAB Outputs

Figure 3: MATLAB Logo

```
function main
    t = linspace(0, 10, 5000);
    dt = t(2) - t(1);

    v = zeros(size(t));
    v(1) = 0;

    for i = 2:length(t)
        v(i) = rk4(v(i-1), t(i-1), dt);
    end

    % Handle for v=0.95
    idx_95 = find(v >= 0.95, 1, 'first');
    if ~isempty(idx_95)
        time_95 = t(idx_95);
        fprintf('Velocity reaches 95%% at %.2f non-dimensional units.\n',
 time_95);
    else
        idx_95 = [];
        time_95 = [];
    end

    % Handle for v=1 (terminal velocity)
    idx_1 = find(v >= 0.999999, 1, 'first');
    if ~isempty(idx_1)
        time_1 = t(idx_1);
        fprintf('Velocity reaches terminal velocity at %.2f non-dimensional
 units.\n', time_1);
    else
        idx_1 = [];
        time_1 = [];
    end

    figure('Position', [100, 100, 800, 480]);
    plot(t, v, 'b-', 'LineWidth', 1.5, 'DisplayName', 'Dimensionless Velocity
 $\hat{V}$');
    xlabel('Time (non-dimensional units)', 'Interpreter', 'latex');
    ylabel('Dimensionless Velocity $\hat{V}$', 'Interpreter', 'latex');
    title('Dimensionless Velocity vs Time', 'Interpreter', 'latex');
    grid on;
    hold on;

    if ~isempty(time_95)
        yline(0.95, 'r--', 'DisplayName', '95% of Terminal Velocity');
        xline(time_95, 'g--', 'DisplayName', sprintf('Time (95%%) = %.2f non-
dimensional units', time_95));
    end

    if ~isempty(time_1)
        yline(1.0, 'm--', 'DisplayName', 'Terminal Velocity');
        xline(time_1, 'Color', [1 0.6 0], 'LineStyle', '--', 'DisplayName',
 sprintf('Time (Terminal) = %.2f non-dimensional units', time_1));
```

```
    end

    legend('show', 'Location', 'best', 'Interpreter', 'latex');
    hold off;
end

function dvdt = model(v)
    dvdt = 1 - v.^2;
end

function v_new = rk4(v, t, dt)
    k1 = dt * model(v);
    k2 = dt * model(v + 0.5 * k1);
    k3 = dt * model(v + 0.5 * k2);
    k4 = dt * model(v + k3);

    v_new = v + (k1 + 2*k2 + 2*k3 + k4) / 6.0;
end
```

```
Velocity reaches 95% at 1.83 non-dimensional units.
Velocity reaches terminal velocity at 7.26 non-dimensional units.
```



Published with MATLAB® R2023a

5 Compiled Plots