Goal: Create a module to calculate the force at each time stamp.

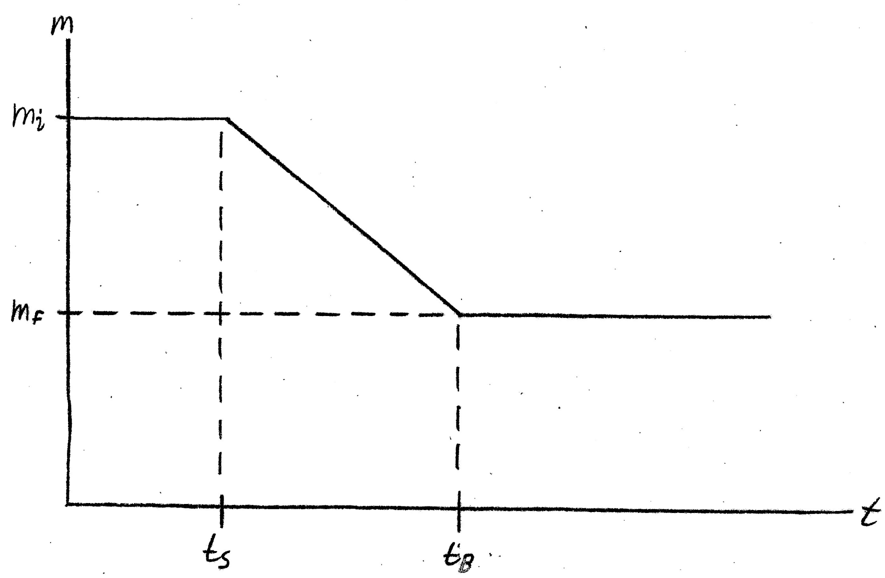
Remember the relationship between force, mass, and acceleration:

$$\vec{F} = m\vec{a}$$

Of course, the mass is also changing with time. The equation should really be written like this:

$$\vec{F}(t) = m(t)\vec{a}(t)$$

Here, each variable is a function of time. The acceleration at each point in time is already known. You will have to figure out a way to calculate mass. Here is how the mass will vary with time:



where m_i is the initial mass, m_f is the final/burnout mass, t_S is the burn start time, and t_B is the burnout time. By inspection:

- for values of $t < t_S$, $m(t) = m_i$
- for values of $t > t_B$, $m(t) = m_f$
- for values of $t_S \leq t \leq t_B$, $m(t) = \frac{m_f - m_i}{t_B - t_S}(t - t_B) + m_i$
  - found using the point-slope formula

Some additional notes:

- m_i and m_f will be given by the user
- t_S will be inferred from when the acceleration values begin to change rapidly. For right now, you can make this an optional parameter with a default value of 1.
- t_B will not be given by the user. The user will give the rocket's burn time (B). Thus, finding t_B is easy: $t_B = t_S + B$

You now have everything you need to create the module. Remember, the function should look something like this:

$$\text{func}=(t,\, \vec{a},\, m_i,\, m_f,\, B,\, t_S = 1)$$

- t is a numpy array: $[t]$
- $\vec{a}$ is a nested numpy array: $[[a_x],\, [a_y],\, [a_z]]$
- m_i, m_f, B, and t_S are all scalar floats