Modelação de Sistemas Físicos - Aula Prática nº3

Problema cap 2.1

In [1]:
```
# Reset ipython kernel
#import os
#os._exit(00)
```

Um carro "A" segue numa estrada à velocidade constante de 70 km/h onde o limite de velocidade é de 50 km/h. Ao passar por um carro patrulha "B", este último parte em sua perseguição com uma aceleração constante de 2 m/s^2.

a) Faça o gráfico da lei do movimento do carro "A" e do carro patrulha "B", $x = x(t)$.

In [2]:
```
# Importar os pacotes "numpy" e "matplotib", e definir respetivos acrónim
import numpy as np
import matplotlib.pyplot as plt
import sympy

# Definir o domínio e resolução da escala do tempo
t = np.linspace(0.0, 0.01, 100)     # Permite definir o número de passos e
#t = np.arange(0.0, 0.01, 0.0001)  # Permite definir o tamanho do passo e
```

Vamos assumir que o carro de patrulha está inicialmente localizado em $x(t=0)=0$ km.

A equação de movimento de movimento do automóvel "A" (com velocidade constante) é:

$$x_{\rm A}(t) = v_{\rm A} t$$

In [3]:
```
# Equação de movimento do carro "A"
v_A = 70.0     # velocidade [km/h]
x_A = v_A * t  # posição [km]
```

A equação de movimento de movimento do carro de patrulha "B" (com aceleração constante) a partir do momento em que passa o carro "A" é:

$$x_\mathrm{B}(t) = \frac{1}{2} a_\mathrm{B} t^2$$

Temos que converter a aceleração do carro de patrulha de $[\mathrm{m/s^2}]$ para $[\mathrm{km/h^2}]$. Ora sabemos que

$[\mathrm{m}] = [\mathrm{km}]/1000$

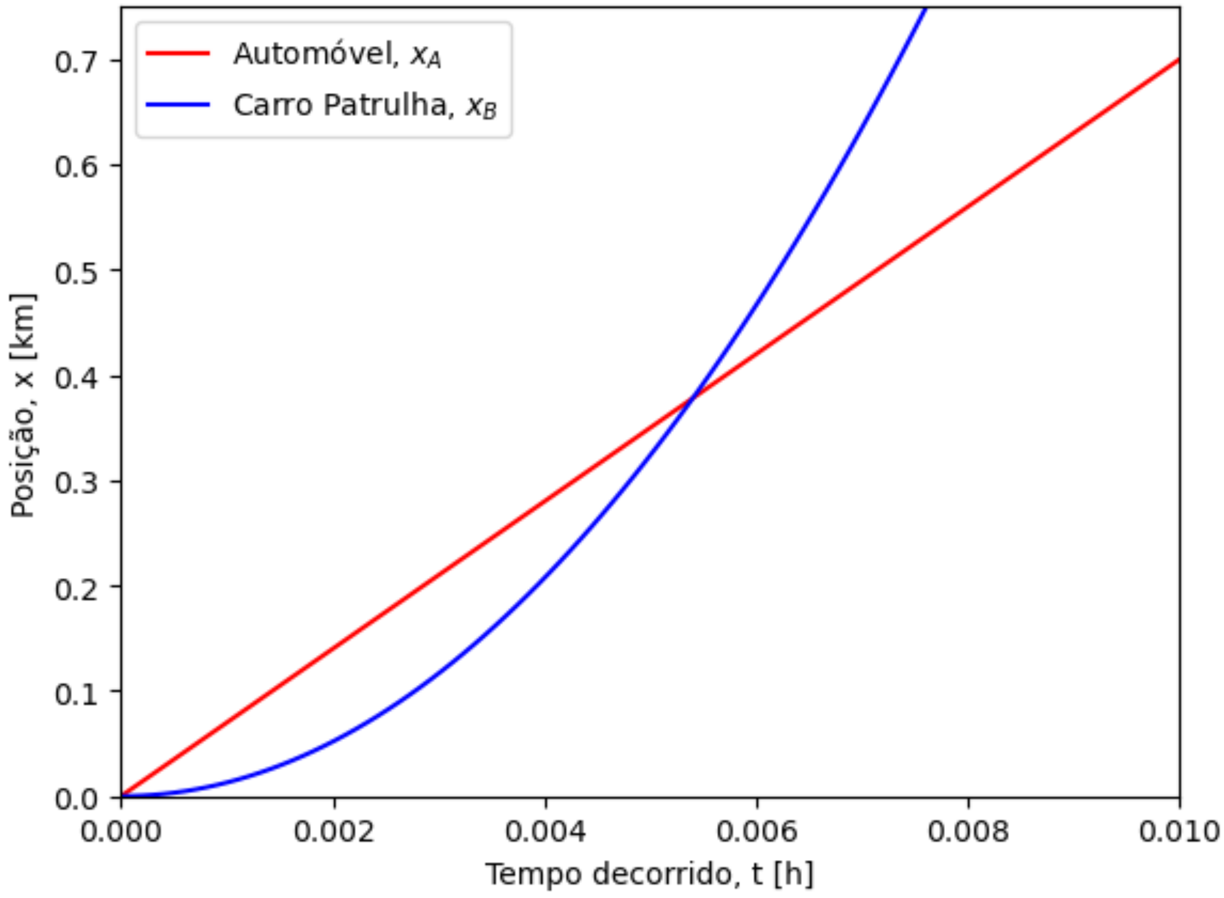
$[\mathrm{s}] = [h]/3600$, ou seja, $[s^2] = [h^2]/3600^2$

e portanto

$[\mathrm{m/s^2}] = ([\mathrm{km}]/1000)/([h^2]/3600^2) = 12960[\mathrm{km/h^2}]$

In [4]:
```
# Equação de movimento do carro de patrulha "P"
a_B = 2.0 * 12960         # aceleração [km/h^2]
x_B = 0.5 * a_B * t ** 2  # posição  [km]

# Representar dados num grafico (usando o matplotlib)
plt.plot(t, x_A, color='red', label=r'Automóvel, $x_A$')
plt.plot(t, x_B, color='blue', label=r'Carro Patrulha, $x_B$')
plt.xlabel("Tempo decorrido, t [h]")
plt.ylabel("Posição, x [km]")
# Definir limites dos eixos do gráfico
plt.xlim(0.0, 0.01)
plt.ylim(0.0, 0.75)
plt.legend()
plt.show()
```



b) Em que instante e qual a distância percorrida pelo carro patrulha quando este último alcança o carro em infração?

Temos que encontrar o valor de t para o qual $x_\mathrm{A}(t) = x_\mathrm{B}(t)$

$v_\mathrm{A} t = \frac{1}{2} a_\mathrm{B} t^2 \Leftrightarrow t = 2 v_\mathrm{A}/a_\mathrm{B} = 0.0054012$ h

Podemos converter o tempo para segundos ($[\mathrm{h}] = 3600 \times [\mathrm{s}]$),

$t = 19.444320$ s

Vamos agora resolver o mesmo problems (obter o tempo decorrido para o encontro dos dois automóveis) usando cálculo simbólico.

Começamos por definir as equações de movimento dos dois automóveis

In [5]:
```
# Definir variáveis independentes
t, v_A, a_B = sympy.symbols('t, v_A, a_B')

x_A = v_A * t            # posição do automóvel "A"
x_B = 1/2 * a_B * t**2   # posição do carro patrulha "B"

# Representar dados num grafico (usando o matplotlib)
tx = sympy.solve(x_A - x_B, t)
print("Encontro ocorre quando t = ", tx, "km")
```

```
Encontro ocorre quando t =  [0.0, 2.0*v_A/a_B] km
```

Em alternativa, podemos substituir os valores da velocidade e aceleração, e resolver a equação novamente,

In [6]:
```
x = sympy.solve((x_A - x_B).subs({v_A: 70.0, a_B: 2.0 * 12960}), t)
print("Encontro ocorre quando x = ", x, "km")
```

```
Encontro ocorre quando x =  [0.0, 0.00540123456790123] km
```

Problema cap 2.2

Um volante de badmington foi largado de uma altura considerável (suficientemente elevada para podermos assumir que atinge um regime estacionário com velocidade terminal constante). A lei do movimento é,

$$y(t) = \frac{v_{\mathrm{T}}^2}{g} \log\left[\cosh\left(\frac{gt}{v_{\mathrm{T}}}\right)\right]$$

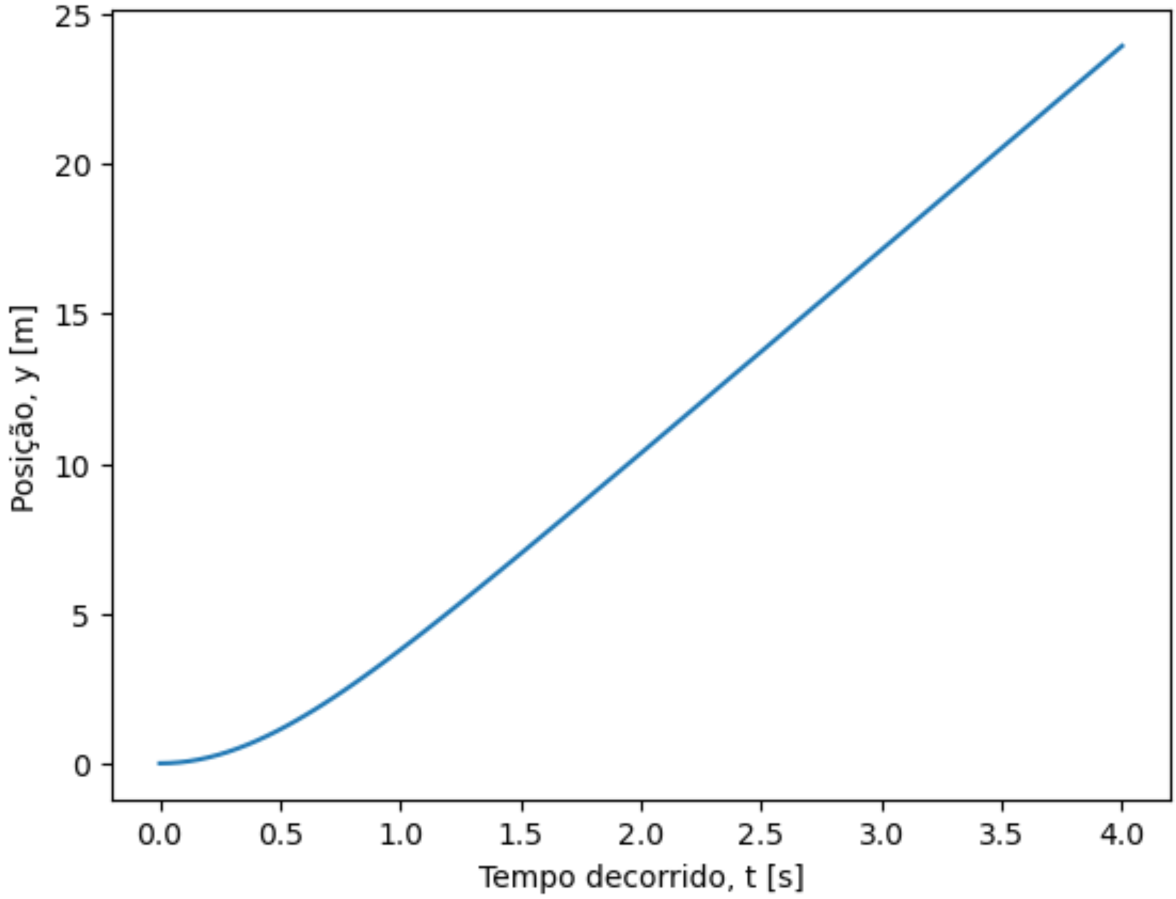
em que a velocidade terminal do volante é $v_{\mathrm{T}} = 6.80$ m/s.

a) Faça o gráfico da lei do movimento $y(t)$ de 0 a 4.0 s.

In [7]:
```
t = np.linspace(0.0, 4.0, 100)    # definir dominio e resolução temporal
v_T = 6.80                        # velocidade terminal do volante [m/s]
g = 9.80665                       # aceleração gravitacional (valor stand

# posição do volante [m]
y = (v_T ** 2 / g) * np.log(np.cosh(g * t / v_T))

# Representar dados num grafico (usando o matplotlib)
plt.plot(t, y)
plt.xlabel("Tempo decorrido, t [s]")
plt.ylabel("Posição, y [m]")
plt.show()
```



b) Determine a velocidade instantânea em função do tempo, usando cálculo simbólico. Faça o gráfico da velocidade em função do tempo de 0 a 4 s, usando o pacote matplotlib.

In [8]:
```
# Definir symbolos
t, v_T, g = sympy.symbols('t, v_T, g')

# Definir expreção da posição do volante
y = (v_T ** 2 / g) * sympy.log(sympy.cosh(g * t / v_T))
```

Sabendo que a velocidade é dada por $v(t) = dx(t)/dt$, podemos usar a função sympy.diff`

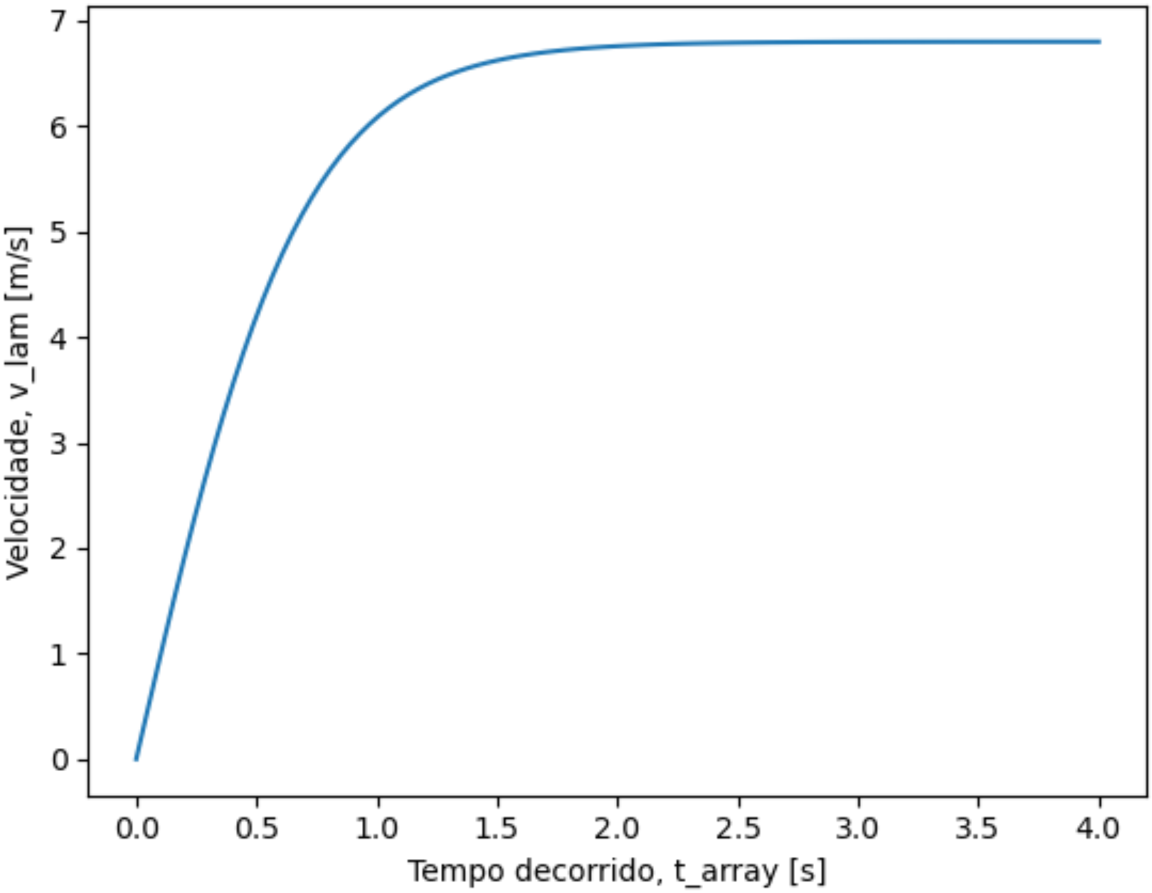
In [9]:
```
# Calcular velocidade instantânea
v = sympy.diff(y, t)
print("v(t) = ", v)
```

```
v(t) =  v_T*sinh(g*t/v_T)/cosh(g*t/v_T)
```

Para calcular o valor de uma expressão simbólica de forma recorrente, usamos o coversor `lambdify`, especificando que o argumento é um array `numpy`,

In [10]:
```
# Converter expressão da velocidade em função de um array numpy
# Nota 1: Só a variável "t" é independente. Temos que especificar todos o
# Nota 2: v_lam é um array numpy se o argumento "t" for um array numpy
v_lam = sympy.lambdify(t, v.subs({v_T:6.80, g: 9.80665}), "numpy")

# Representar dados num grafico (usando o matplotlib)
t_array = np.linspace(0.0, 4.0, 100)
plt.plot(t_array, v_lam(t_array))
plt.xlabel("Tempo decorrido, t_array [s]")
plt.ylabel("Velocidade, v_lam [m/s]")
plt.show()
```



c) Determine a aceleração instantânea em função do tempo, usando cálculo simbólico. Faça o gráfico da aceleração em função do tempo de 0 a 4 s, usando o pacote matplotlib.

Sabendo que a aceleração é dada por $a(t) = dv(t)/dt$, podemos usar novamente a função `sympy.diff`

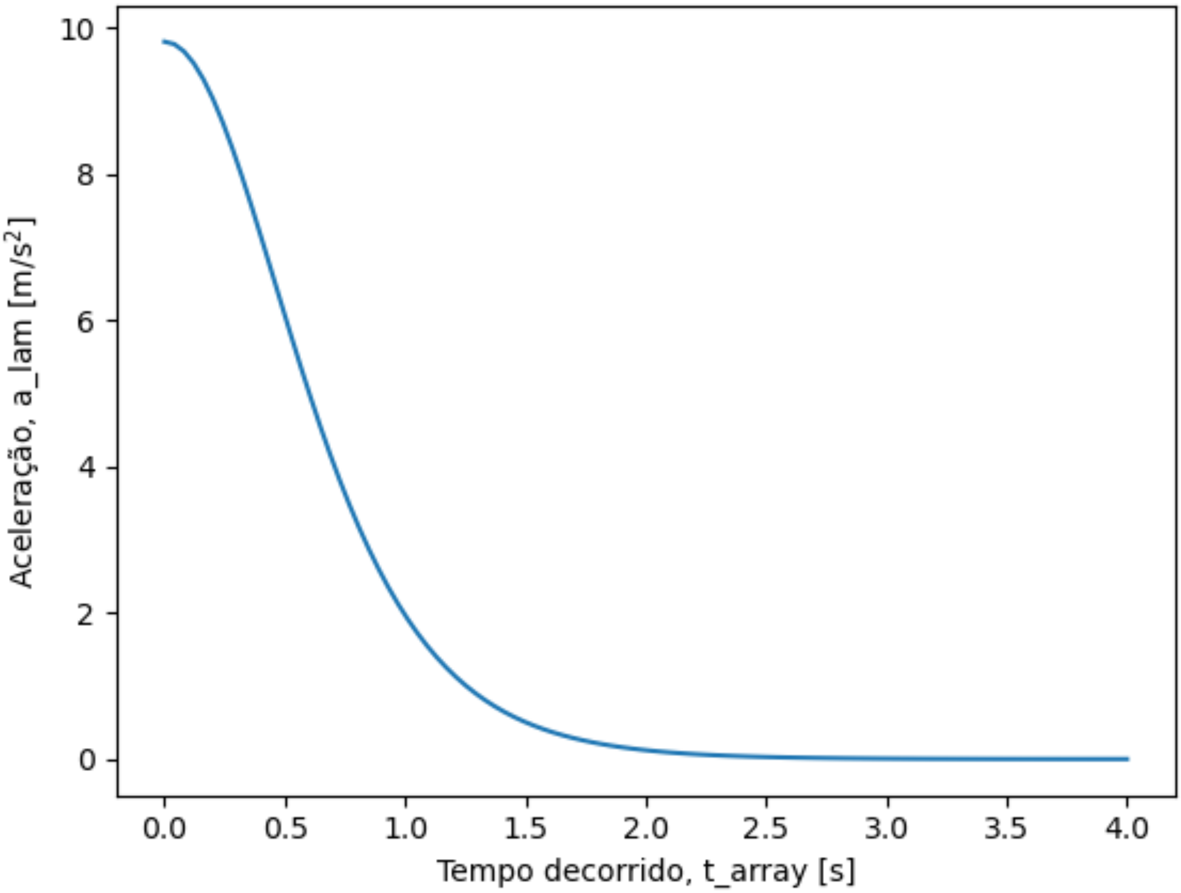
In [11]:
```
# Calcular velocidade instantânea
a = sympy.diff(v, t)
print("a(t) = ", a)
```

```
a(t) =  -g*sinh(g*t/v_T)**2/cosh(g*t/v_T)**2 + g
```

Usamos novamente o coversor `lambdify`, para obter os valores da aceleração num array `numpy`.

In [12]:
```python
# Converter expressão da aceleração em função de um array numpy
# Nota 1: Só a variável "t" é independente. Temos que especificar todos o
# Nota 2: a_lam é um array numpy se o argumento "t" for um array numpy
a_lam = sympy.lambdify(t, a.subs({v_T:6.80, g: 9.80665}), "numpy")

# Representar dados num grafico (usando o matplotlib)
t_array = np.linspace(0.0, 4.0, 100)
plt.plot(t_array, a_lam(t_array))
plt.xlabel("Tempo decorrido, t_array [s]")
plt.ylabel(r"Aceleração, a_lam [m/s$^2$]")
plt.show()
```



d) Mostre que a aceleração é dada por

$$a(t) = g\left[1 - \left(\frac{v}{v_\mathrm{T}}\right)^2\right]$$

Pelo resultado da alíniea b) sabemos que

$$v(t) = v_\mathrm{T}\frac{\sinh(gt/v_\mathrm{T})}{\cosh(gt/v_\mathrm{T})} \Leftrightarrow \frac{\sinh(gt/v_\mathrm{T})}{\cosh(gt/v_\mathrm{T})} = v/v_\mathrm{T}$$

Pelo resultado da alíniea c) sabemos que

$$a(t) = g - g\left[\frac{\sinh(gt/v_\mathrm{T})}{\cosh(gt/v_\mathrm{T})}\right]^2$$

Portanto, substituindo a expressão de $v(t)$ na expressão de $a(t)$ obtemos

$$a(t) = g\left[1 - \left(\frac{v}{v_\mathrm{T}}\right)^2\right]$$

e) Se o volante for largado de uma altura de 20 m, quanto tempo demora a atingir o solo? Compare com o tempo que demoraria se não houvesse resistência do ar.

In [13]:
```
#sympy.solve((20 - y).subs({v_T: 6.80, g: 9.80665}), t)
t20_ar = sympy.nsolve((20 - y).subs({v_T: 6.80, g: 9.80665}), t, 0.0)
print("Com resistência do ar, o volante atinge o solo quando t = ", t20_a
```

```
Com resistência do ar, o volante atinge o solo quando t =  3.4217737250722
3 s
```

Se não houvesse resitência do ar, a posição seria $y_\mathrm{vac}(t) = \frac{1}{2}gt^2$. Assim

In [14]:
```
# Posição do volante no vácuo (sem resistência do ar)
y_vac = 1/2 * g * t**2

t20_vac = sympy.nsolve((20 - y_vac).subs(g, 9.80665), t, 0.0)
print("Sem resistência do ar, o volante atinge o solo quando t = ", t20_v
```

```
Sem resistência do ar, o volante atinge o solo quando t =  2.0196199771025
5 s
```

f) Nas condições da alínea anterior, qual o valor da velocidade e da aceleração quando o volante chega ao solo?

In [15]:
```
v_vac = sympy.diff(y_vac,t)
print(v_vac)
print(v_vac.subs({g: 9.80665, t: 2.01961997710255}), "m/s")
```

```
1.0*g*t
19.8057062484527 m/s
```

In [16]:
```
a_vac=sympy.diff(v_vac,t)
print(a_vac)
print(a_vac.subs({g: 9.80665}), "m/s^-2")
```

```
1.0*g
9.80665000000000 m/s^-2
```