Chapter 13 Initial-Value Problems for Linear Differential Equations

Initial values problems get rid of the arbitary constant in favor of a specific equation to fit the initial conditions. The initial condition coordinates are both defined at the same point of the domain line. After the ODE has been solved, at least one of the initial conditions can be used to verify, or at least support, the new solution.

Cutting and pasting and Wolfram Alpha. Wolfram Alpha is amenable to accepting pasted entries. In this chapter pastable expressions are given a distinctive boundary fence, exemplified by the sample: !| abcdef |!

In the above pseudo-entry, only the alpha characters would be copied for transfer to Wolfram Alpha.

13.1 Solve $y'' - y' - 2y = 4x^2$ $y(0) = 1, y'(0) = 4$

The entry is made into Wolfram Alpha:

!| y'' - y' - 2 * y = 4 * x^2 , y(0) = 1, y'(0) =4) |!

and the answer is received:

$$y(x) = -2x^2 + 2x + 2e^{-x} + 2e^{2x} - 3$$

13.2 Solve $y'' - y' - 2y = 4x^2$ $y(0) = 1, y'(0) = 4$

The entry is made into Wolfram Alpha:

!| y'' - 2 * y' + y = e^x/x , y(1) = 0, y'(1) = 1 |!

and the answer is received:

$$y(x) = e^{x-1}(-ex + x + e\,x\log(x) + e - 1)$$

13.3 Solve $y'' + 4y' + 8y = \sin x;$ $y(0) = 1, y'(0) = 0$

The entry is made into Wolfram Alpha:

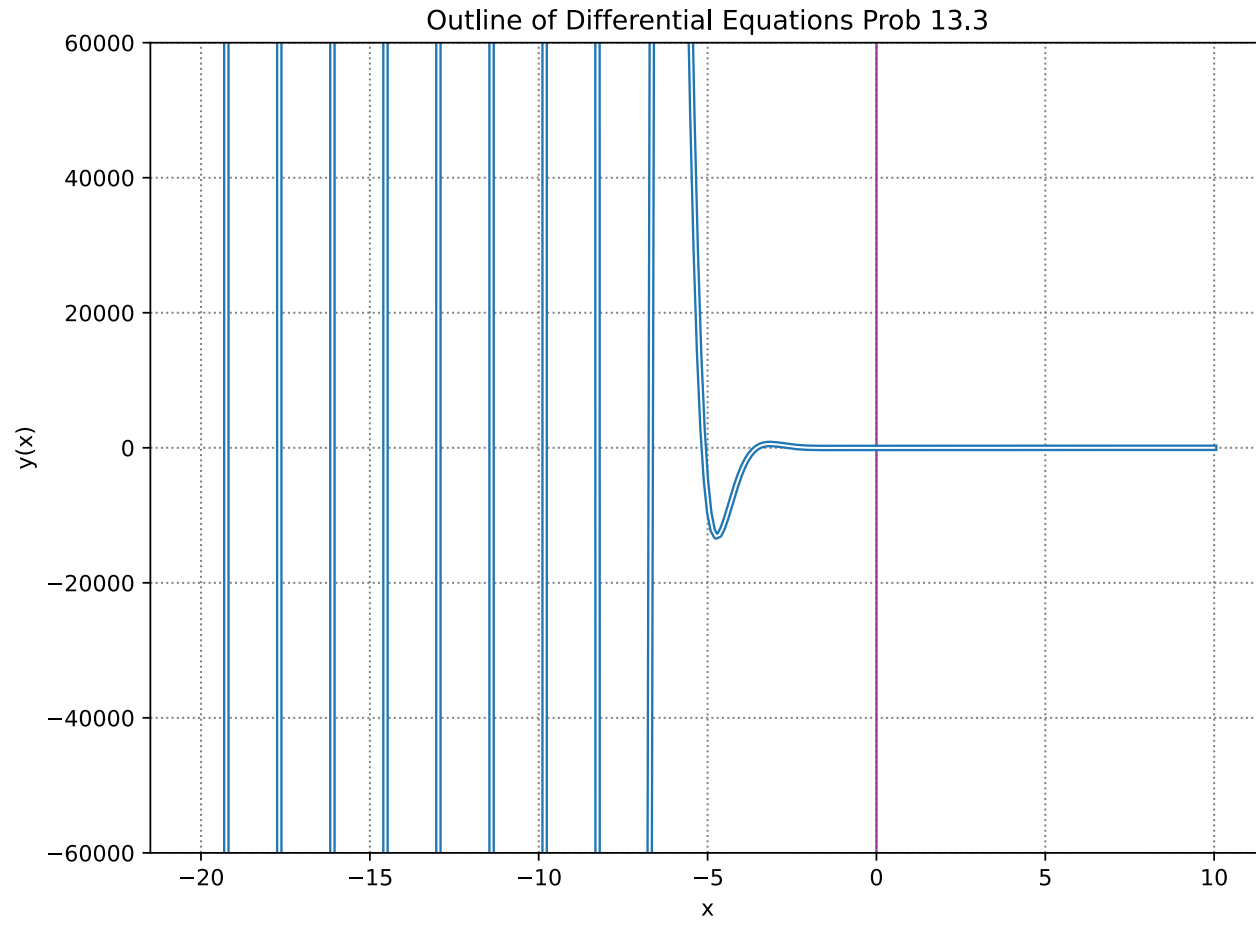
!| d^2y/dx^2 + 4 * y' + 8 * y = sin(x), y(0) = 1, y'(0) = 0 |!

and the answer is received:

$$y(x) = \frac{1}{65}e^{-2x}(7e^{2x}\sin(x) + 69\cos(2x) + (131\sin(x) - 4e^{2x})\cos(x))$$

The Wolfram Alpha answer and the text answer are not easily reconciled. The plot below helps vouch for equivalence between the two, especially considering the large vertical scale.

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

x = np.linspace(-20,10,300)
y3 = (1/65)*np.exp(-2*x)*(7*np.exp(2*x)*np.sin(x)+ 69 * np.cos(2*x) + \
            (131*np.sin(x)-4*np.exp(2*x))*np.cos(x))
y4 = np.exp(-2*x)*((69/65)*np.cos(2*x) + (131/130)*np.sin(2*x)) + \
            (7/65)*np.sin(x) - (4/65)*np.cos(x)


plt.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
plt.xlabel("x")
plt.ylabel("y(x)")
plt.title("Outline of Differential Equations Prob 13.3")
plt.rcParams['figure.figsize'] = [9, 7.5]


ax = plt.gca()
#ax.axhline(y=0, color='#993399', linewidth=1)
ax.axvline(x=0, color='#993399', linewidth=1)
ratio = 0.0002
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)


#plt.text(965, -2.5e43, "-np.log(abs((np.cos(x/2))**2 - (np.sin(x/2))**2)))
#plt.text(965, 2.25e43, "SOLN: y = 3*np.exp(x**2) + 1/2", size=10,\
        # bbox=dict(boxstyle="square", ec=('#8C564B'),fc=(1., 1., 1),))
plt.ylim(-60000,60000)
plt.plot(x, y3, linewidth = 3)
plt.plot(x, y4, linewidth = 0.9, color = 'w')
plt.show()
```



13.4 Solve $y''' - 6y'' + 11y' - 6y = 0$; $y(\pi) = 0, y'(\pi) = 0, y''(\pi) = 1$

The entry is made into Wolfram Alpha:

!| y''' - 6 * y'' + 11 * y' - 6 * y = 0, y(pi) = 0, y'(pi) = 0, y''(pi) =1 |!

and the answer is received:

$$y(x) = \frac{1}{2} e^{x-3\pi} (e^{\pi} - e^{x})^2$$

The Wolfram Alpha answer and the text answer are equivalent.

13.5 Solve $\ddot{x} + 4x = \sin^2 2t$; $x(0) = 0, \dot{x}(0) = 0$

The entry is made into Wolfram Alpha:

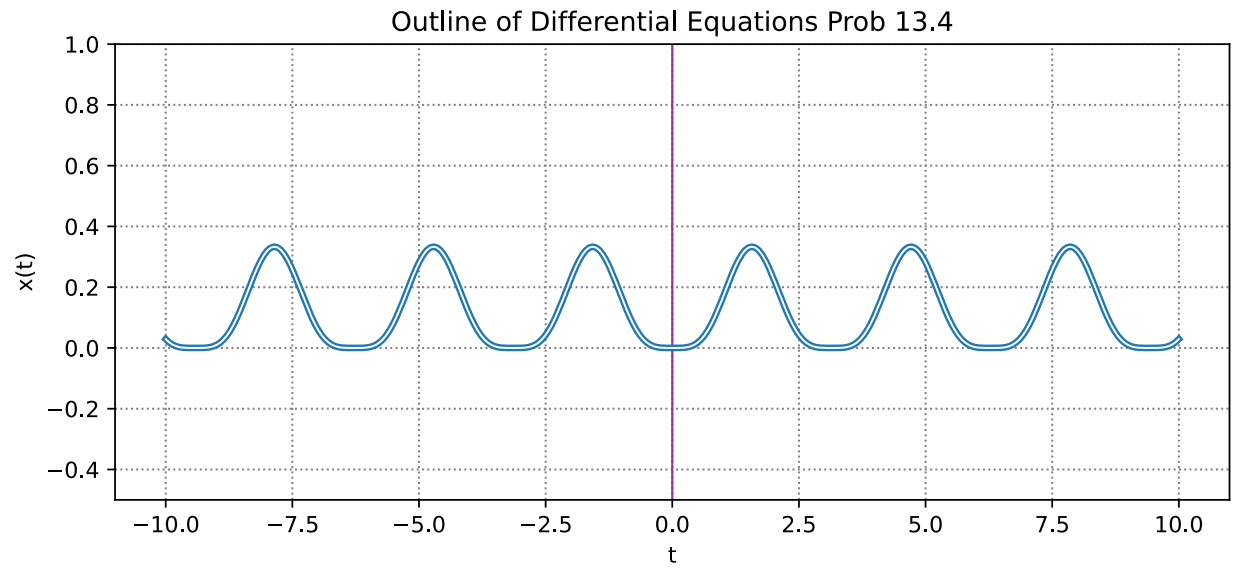
!| x'' + 4 * x = sin^2(2 * t), x(0) = 0, x'(0) = 0 |!

and the answer is received:

$$x(t) = \frac{\sin^4 t}{3}$$

The Wolfram Alpha answer and the text answer are equivalent, though they don't look like it. It takes a superimposed plot like the one below to convince that they are.

In [72]:
```
import numpy as np
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

x = np.linspace(-10,10,300)
y3 = ((np.sin(x))**4)/3
y4 = -(1/6)*np.cos(2*x) + (1/6)*(np.cos(2*x))**2 + (1/12)*(np.sin(2*x))**2


plt.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
plt.xlabel("t")
plt.ylabel("x(t)")
plt.title("Outline of Differential Equations Prob 13.4")
plt.rcParams['figure.figsize'] = [9, 4]


ax = plt.gca()
#ax.axhline(y=0, color='#993399', linewidth=1)
ax.axvline(x=0, color='#993399', linewidth=1)
ratio = 6
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)


#plt.text(965, -2.5e43, "-np.log(abs((np.cos(x/2))**2 - (np.sin(x/2))**2)))
#plt.text(965, 2.25e43, "SOLN: y = 3*np.exp(x**2) + 1/2", size=10,\
        # bbox=dict(boxstyle="square", ec=('#8C564B'),fc=(1., 1., 1),))
plt.ylim(-0.5,1)
plt.plot(x, y3, linewidth = 3)
plt.plot(x, y4, linewidth = 0.9, color = 'w')
plt.show()
```



13.6 Solve $\ddot{x} + 4x = \sin^2 2t; \qquad x(\frac{\pi}{8}) = 0,\ \dot{x}(\frac{\pi}{8}) = 0$

The entry is made into Wolfram Alpha:

!| x'' + 4 * x = sin^2(2 * t), x(pi/8) = 0, x'(pi/8) = 0 |!

and the answer is received:

$$x(t) = \frac{1}{48}(-\sqrt{2}\sin(2t) - 5\sqrt{2}\cos(2t) + 2\cos(4t) + 6)$$

And because of some suspicion in the results, another method was passed to W|A for execution:

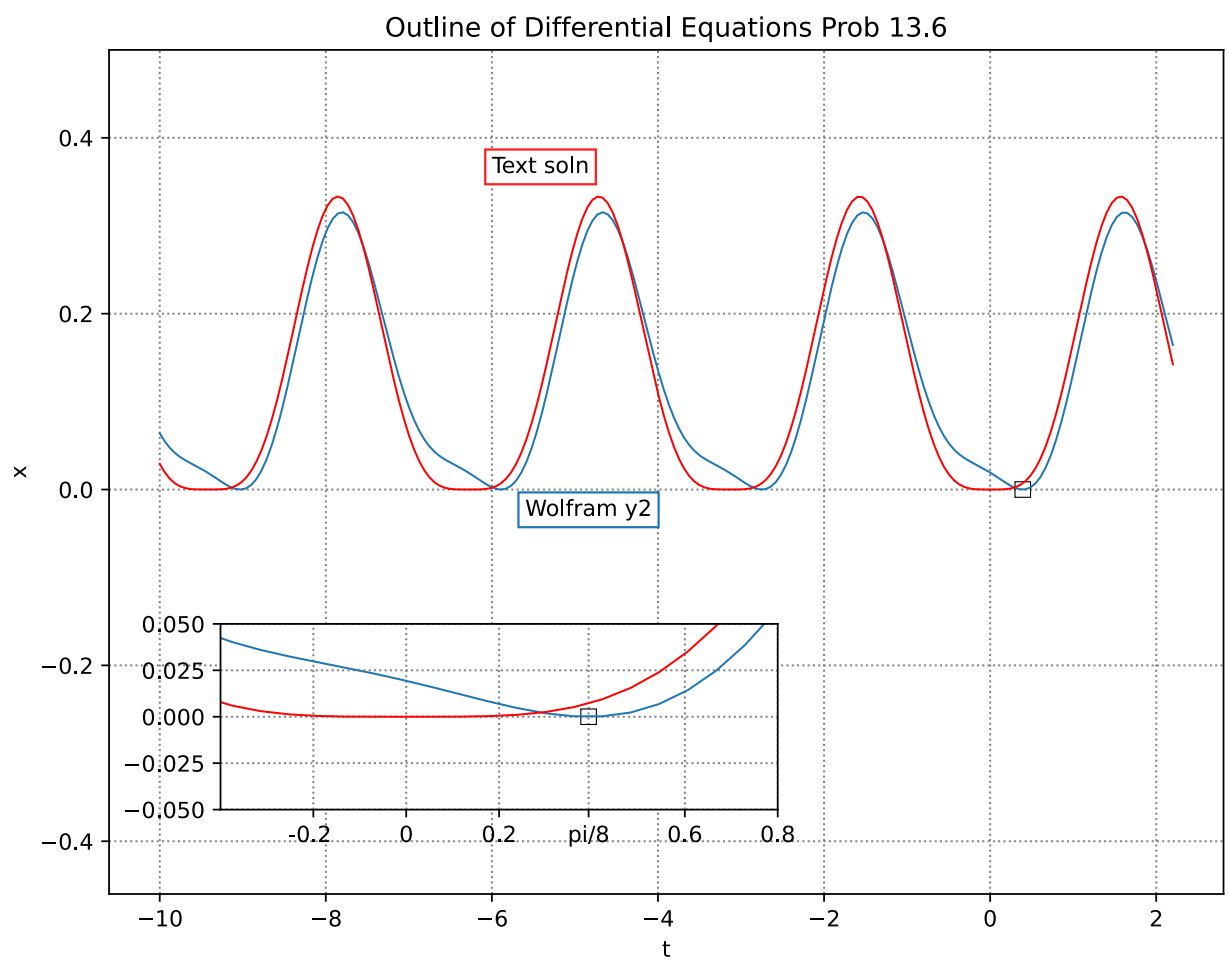
!| solve {x'' + 4 x = sin^2(2 * t), x(pi/8) = 0, x'(pi/8) = 0} using r k f algorithm |!

The page layout for the above method includes a section with the heading "exact solution of equation" which is different than the solution retrieved at the first instigation.

The Wolfram Alpha answer and the text answer are similar, but not close. Comparison plot is shown below. Apparently there is a sizable discrepancy. The functions $y2$ and $y3$ are the two corresponding to Wolfram Alpha. The text answer plot seems more regular and a closer fit to the previous problem, which is related to this one. However, close examination of the first initial condition coordinate throws support toward Wolfram Alpha.

In [3]:
```
import numpy as np
import matplotlib.pyplot as pl
import math as ma

%config InlineBackend.figure_formats = ['svg']

x = np.linspace(-10.,2.2,200)
y2 = (1/48)*(3*np.sin(2*x)*np.sin(2*x) - np.sqrt(2)*np.sin(2*x) -
    np.sin(2*x)*np.sin(6*x) + 9*np.cos(2*x)*np.cos(2*x) - \
    np.cos(6*x)*np.cos(2*x) - 5*np.sqrt(2)*np.cos(2*x))
x1 = np.linspace(1.0,3,200)
y3 = (1/48)*(-np.sqrt(2)*np.sin(2*x) -5*np.sqrt(2)*np.cos(2*x) + \
    2*np.cos(4*x) + 6)
y4 = -(1/6)*np.cos(2*x) + (1/6)*(np.cos(2*x))**2 + (1/12)*(np.sin(2*x))**2

plt.rcParams['figure.figsize'] = [9, 7]

fig, ax = plt.subplots()
axin1 = ax.inset_axes([0.1, 0.1,
                       0.5, 0.22])

axin1.set(xlim=(-0.4, 0.5), ylim=(-.05, .05))
axin1.set_xticks([-0.2, 0, 0.2,   ma.pi / 8, 0.6, 0.8], labels=\
    ['-0.2 ','0','0.2', 'pi/8', '0.6', '0.8'])
axin1.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
ax.grid(True, linestyle='dotted', color='gray', linewidth=0.9)
plt.title('Outline of Differential Equations Prob 13.6')
plt.xlabel('t')
plt.ylabel('x')

ax.text(-5.6, -0.03, "Wolfram y2", size=10,bbox=dict\
         (boxstyle="square", ec=('#1F77B4'),fc=(1., 1., 1),))
ax.text(-6, 0.36, "Text soln", size=10,bbox=dict\
         (boxstyle="square", ec=('#FF1D1D'),fc=(1., 1., 1),))
apts = np.array([ma.pi/8])
bpts = np.array([0])
ax.plot(apts, bpts, markersize=7, color='k', marker='s', mfc='none', \
    markeredgewidth=0.5)
axin1.plot(apts, bpts, markersize=7, color='k', marker='s', mfc='none', \
    markeredgewidth=0.5)
axin1.labelsize = 5
plt.ylim(-0.46,0.5)

ax.plot(x, y2,  linewidth = 0.9)
#ax.plot(x, y3, linewidth = 0.9)
ax.plot(x, y4,  color= 'r', linewidth = 0.9)
axin1.plot(x, y2, linewidth = 0.9)
#axin1.plot(x, y3, linewidth = 0.9)
axin1.plot(x, y4, color = 'r', linewidth = 0.9)
```

Out[3]: [<matplotlib.lines.Line2D at 0x7fd4d814a670>]



In [ ]: