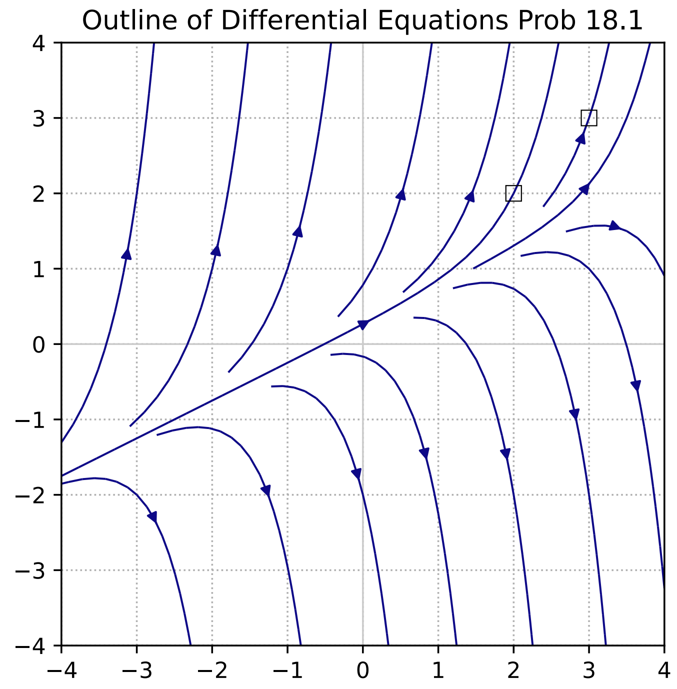
Chapter 18 Graphical and Numerical Methods for Solving 1st Order Differential Equations. Not as exact as closed form solutions, but necessary for real-world applications, and for getting a larger view of the problem to balance other solution methods.

18.1 Construct a direction field for the differential equation $y' = 2y - x$.

Direction fields are undoubtedly useful as a visual aid, but a stream plot looks less cluttered, and may give better information.

In [35]:
```
import numpy as np
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

# Creating dataset
w = 4
Y, X = np.mgrid[-w:w:100j, -w:w:100j]
U = np.ones_like(X) #dxdt = 1
V = 2*Y - X
speed = np.sqrt(U**2 + V**2)

seek_points = np.array([[2,3.5,3,-2,-1,-3,  1,     2, 3, \
                         -1.5, 0, -3, 0.25, 3.5, 1.75, 3],
                        [2,3,  1, 1, 1,-2, -2.25, -2,-2, \
                         -1.5,-2,  2, 1.25, 1.5, 3,    3]])

fig, ax = plt.subplots()
ax.grid(True, which='both', linestyle='dotted')
ax.axhline(y=0, color='0.8', linewidth=0.8)
ax.axvline(x=0, color='0.8', linewidth=0.8)

ratio = 1.0
#ratio is adjusted by eye to get squareness of x and y spacing
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)


strm = ax.streamplot(X, Y, U, V, color = U,
                     linewidth = 0.9,
                     cmap ='plasma',
                     start_points = seek_points.T)

plt.title("Outline of Differential Equations Prob 18.1")
plt.rcParams['figure.figsize'] = [5, 5]

xpts = np.array([2, 3])
ypts = np.array([2, 3])
plt.plot(xpts, ypts, markersize=7, color='k', marker='s', \
         mfc='none', linestyle = 'none', markeredgewidth=0.5)
plt.show()
```



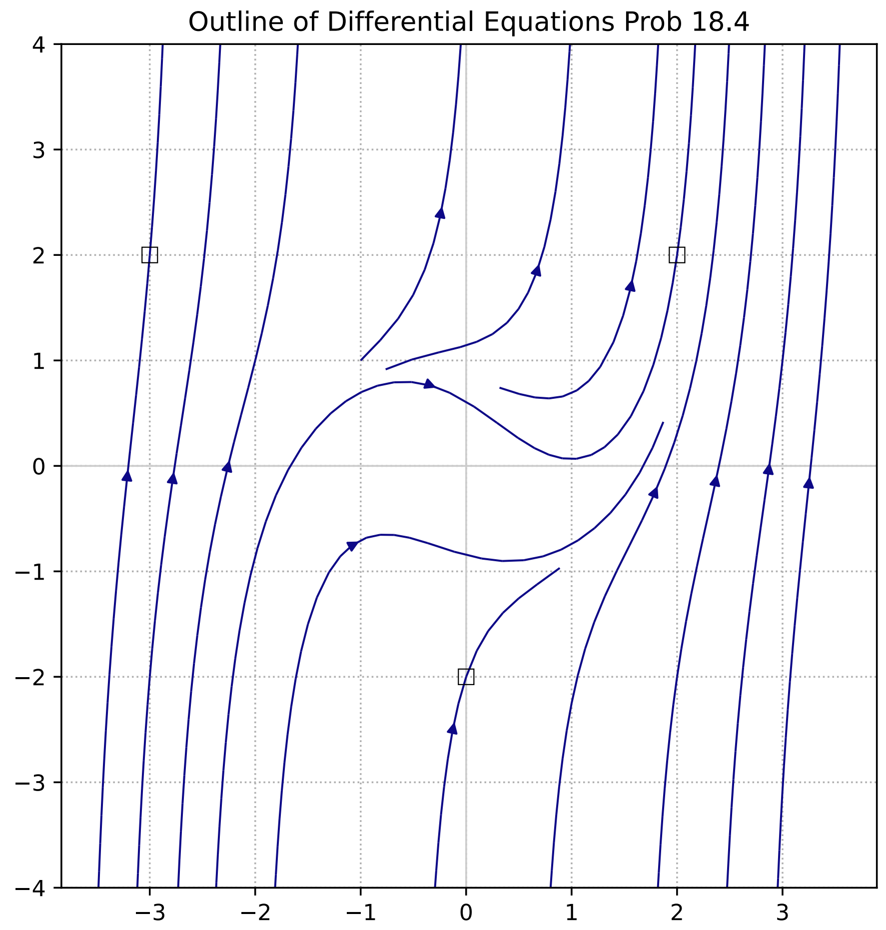
18.3 Draw two solution curves to the differential equation given in Problem 18.1.

Two solution curves, coming right up. The ones shown on the above plot intersect the coordinates [2, 2] and [3, 3].

18.4 Construct a direction field for the differential equation $y' = x^2 + y^2 - 1$.

The next equation sounds suspiciously like a circle. However, at the scale and resolution viewed in the plot below, it is not a circle. Interestingly, the seek points can remain exactly the same as the last plot and still give an accurate impression of the equation environment.

In [139]:
```
import numpy as np
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

# Creating dataset
w = 4
Y, X = np.mgrid[-w:w:100j, -w:w:100j]
U = np.ones_like(X) #dxdt = 1
V = X**2 + Y**2 -1
speed = np.sqrt(U**2 + V**2)

seek_points = np.array([[2,3.5,3,-2,-1,-3,  1,     2, 3, \
                         -1.5, 0, -3, 0.25, 3.5, 1.75, 3],
                        [2,3,  1, 1, 1,-2, -2.25, -2,-2, \
                         -1.5,-2,  2, 1.25, 1.5, 3,    3]])

fig, ax = plt.subplots()
ax.grid(True, which='both', linestyle='dotted')
ax.axhline(y=0, color='0.8', linewidth=0.8)
ax.axvline(x=0, color='0.8', linewidth=0.8)

ratio = 1.0
#ratio is adjusted by eye to get squareness of x and y spacing
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)


strm = ax.streamplot(X, Y, U, V, color = U,
                     linewidth = 0.9,
                     cmap ='plasma',
                     start_points = seek_points.T)

#ax.set_title('Markers at (2,2) and (3,3)')
plt.title("Outline of Differential Equations Prob 18.4")
plt.rcParams['figure.figsize'] = [5, 5]

xpts = np.array([-3, 0, 2])
ypts = np.array([2, -2, 2])
plt.plot(xpts, ypts, markersize=7, color='k', marker='s', \
         mfc='none', linestyle= 'none', markeredgewidth=0.5)
plt.show()
```



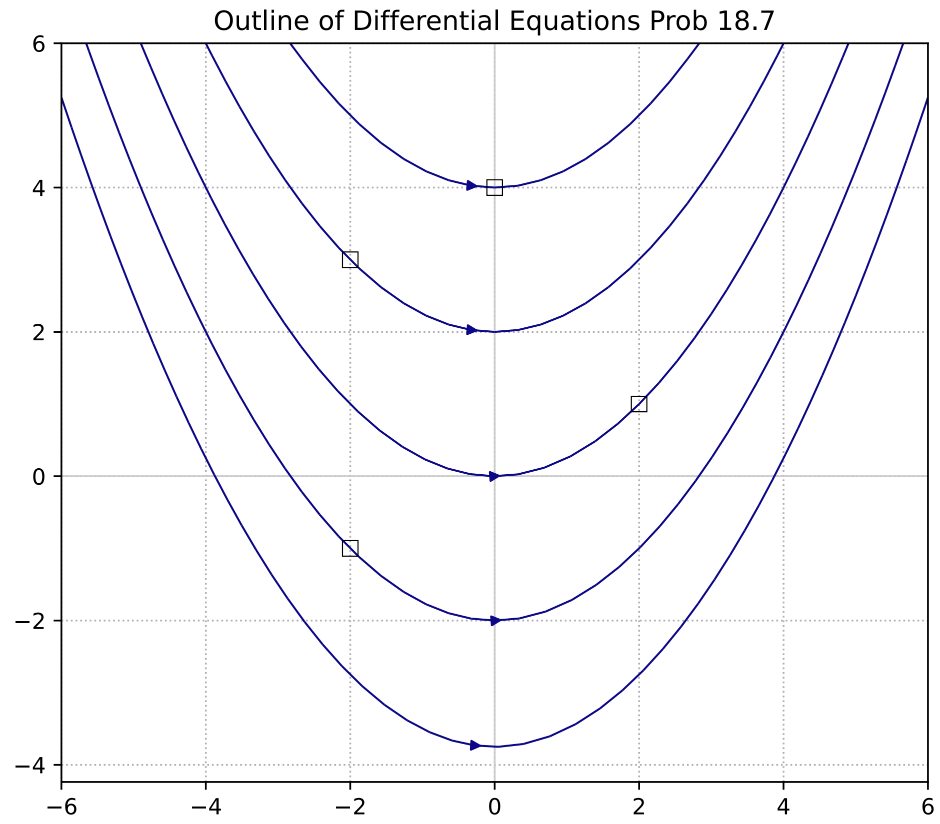
18.6 Draw three solution curves to the differential equation given in Problem 18.4.

Since the stream plot is constructed based on the seek points, representative curves are guaranteed to pass through them.

18.7 Construct a direction field for the differential equation $y' = x/2$.

The solution to the ODE plotted is a family of parabolas.

In [138]:
```
import numpy as np
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

# Creating dataset
w = 6
Y, X = np.mgrid[-w:w:100j, -w:w:100j]
U = np.ones_like(X) #dxdt = 1
V = X/2
speed = np.sqrt(U**2 + V**2)

seek_points = np.array([[2, 0, 4, 0, 0, 5,   -2 ],
                        [1, 0, 2, 2, 4,  2.5, -1]])

fig, ax = plt.subplots()
ax.grid(True, which='both', linestyle='dotted')
ax.axhline(y=0, color='0.8', linewidth=0.8)
ax.axvline(x=0, color='0.8', linewidth=0.8)

ratio = 1.0
#ratio is adjusted by eye to get squareness of x and y spacing
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)


strm = ax.streamplot(X, Y, U, V, color = U,
                     linewidth = 0.9,
                     cmap ='plasma',
                     start_points = seek_points.T)

plt.title("Outline of Differential Equations Prob 18.7")
plt.rcParams['figure.figsize'] = [7, 7]

xpts = np.array([[2], [-2], [0], [-2]])
ypts = np.array([[1], [3], [4], [-1]])
plt.plot(xpts, ypts, markersize=7, color='k', marker='s', \
         mfc='none', linestyle ='none', markeredgewidth=0.5)
plt.show()
```

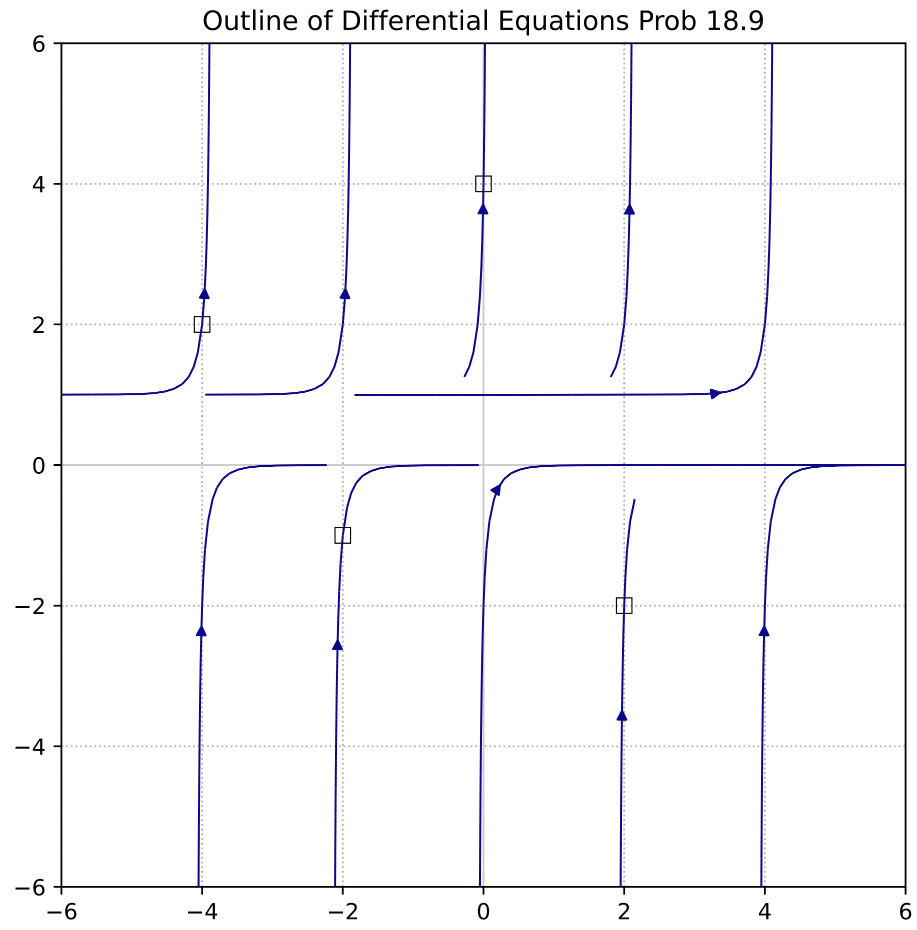


18.8 Draw four solution curves to the differential equation given in Problem 18.7.

Four of the five curves shown are solution curves.

18.9 Draw solution curves to the differential equation $y' = 5y(y-1)$.

In [123]:
```
import numpy as np
import matplotlib.pyplot as plt

%config InlineBackend.figure_formats = ['svg']

# Creating dataset
w = 6
Y, X = np.mgrid[-w:w:100j, -w:w:100j]
U = np.ones_like(X) #dxdt = 1
V = 5*Y*(Y - 1)
speed = np.sqrt(U**2 + V**2)

seek_points = np.array([[-4, -2, 4, 0, 2, 0 ,  2,  4, -2, -4],
                        [  2,  2, 2, 4, 2, -2, -2, -2, -1, -2]])

fig, ax = plt.subplots()
ax.grid(True, which='both', linestyle='dotted')
ax.axhline(y=0, color='0.8', linewidth=0.8)
ax.axvline(x=0, color='0.8', linewidth=0.8)

ratio = 1.0
#ratio is adjusted by eye to get squareness of x and y spacing
xleft, xright = ax.get_xlim()
ybottom, ytop = ax.get_ylim()
ax.set_aspect(abs((xright-xleft)/(ybottom-ytop))*ratio)


strm = ax.streamplot(X, Y, U, V, color = U,
                     linewidth = 0.9,
                     cmap ='plasma',
                     start_points = seek_points.T)

plt.title("Outline of Differential Equations Prob 18.9")
plt.rcParams['figure.figsize'] = [7, 7]

xpts = np.array([-4, 2, 0, -2])
ypts = np.array([2, -2, 4, -1])
plt.plot(xpts, ypts, markersize=7, color='k', marker='s', \
         mfc='none', linestyle = 'none', markeredgewidth=0.5)
plt.show()
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: