Untitled

April 30, 2018

1 Phys 116B HW 3

1.1 Imports

In [195]:
```
import matplotlib.pyplot as plt
from scipy import *
from scipy import integrate
from scipy.integrate import odeint
import numpy as np
%matplotlib inline
```

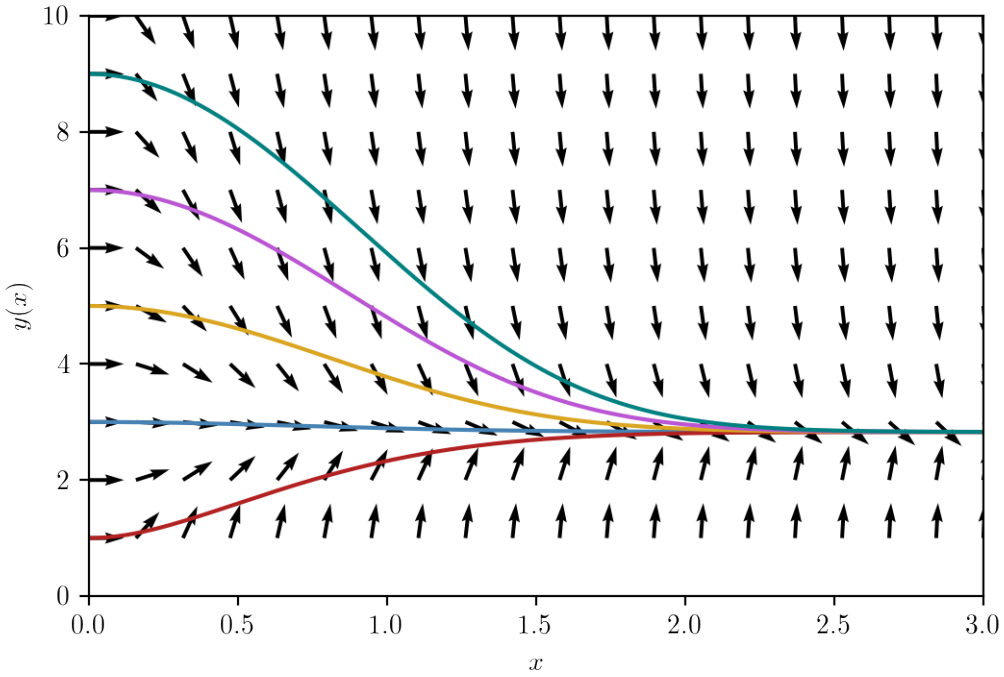
1.2 Problem 3

In [319]:
```
## Vector field function
def dydt(y, t):
    return t * (8. - y**2) / y
```

In [320]:
```
ics = np.array([[0., 3.], [0., 1.], [0., 5.], [0., 7.], [0., 9.]])

n_pts = 200
n_ics = ics.shape[0]

t_final = 3.

ys = np.arange(n_ics * n_pts, dtype=float).reshape(n_ics, n_pts)
ts = np.arange(n_ics * n_pts, dtype=float).reshape(n_ics, n_pts)

ts = np.array([np.linspace(ics[i, 0], t_final, num = n_pts)
               for i in range(n_ics)])


ys[0] = odeint(dydt, ics[0, 1], ts[0]).reshape(n_pts)
ys[1] = odeint(dydt, ics[1, 1], ts[1]).reshape(n_pts)
ys[2] = odeint(dydt, ics[2, 1], ts[2]).reshape(n_pts)
ys[3] = odeint(dydt, ics[3, 1], ts[3]).reshape(n_pts)
ys[4] = odeint(dydt, ics[4, 1], ts[4]).reshape(n_pts)
```

In [321]:
```
fig = plt.figure(dpi=200)
ax = fig.add_subplot(111)
```

```
ax.plot(ts[0], ys[0])
ax.plot(ts[1], ys[1])
ax.plot(ts[2], ys[2])
ax.plot(ts[3], ys[3])
ax.plot(ts[3], ys[4])

Xs = np.linspace(0., t_final, num=20)
Ys = np.linspace(1., 10., num=10)

X, Y = np.meshgrid(Xs, Ys)
U = 1.
V = X * (8. - Y**2) / Y
N = np.sqrt(U**2 + V**2)

U2, V2 = U / N, V / N
ax.quiver(X, Y, U2, V2)

plt.ylim([0., 10.])
plt.xlim([0., t_final])
plt.ylabel(r'$y(x)$')
plt.xlabel(r'$x$')
plt.savefig("latex/files/prob_3.pdf")
plt.show()
```



1.3 Problem 2

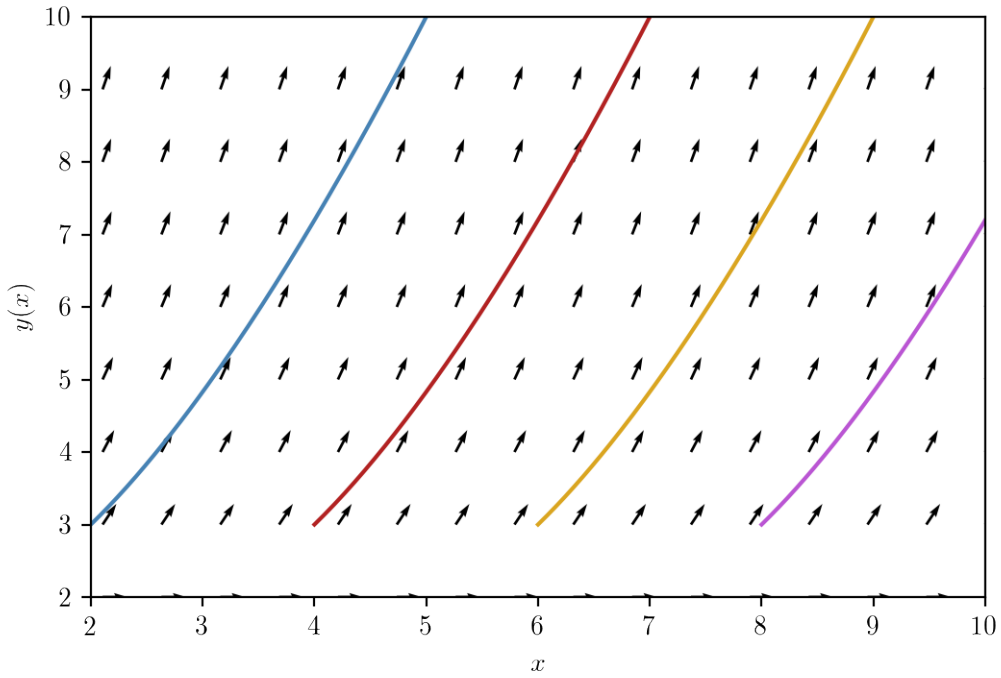
In [313]:
```
## Vector field function
def dydt(y, t):
    return 1.5 * (y - 2.)**(1. / 3.)
```

In [314]:
```
ics = np.array([[2., 3.], [4., 3.], [6., 3.], [8., 3.]])

n_pts = 200
n_ics = ics.shape[0]

t_final = 10.

ys = np.arange(n_ics * n_pts, dtype=float).reshape(n_ics, n_pts)
ts = np.arange(n_ics * n_pts, dtype=float).reshape(n_ics, n_pts)

ts = np.array([np.linspace(ics[i, 0], t_final, num = n_pts)
               for i in range(n_ics)])


ys[0] = odeint(dydt, ics[0, 1], ts[0]).reshape(n_pts)
ys[1] = odeint(dydt, ics[1, 1], ts[1]).reshape(n_pts)
ys[2] = odeint(dydt, ics[2, 1], ts[2]).reshape(n_pts)
ys[3] = odeint(dydt, ics[3, 1], ts[3]).reshape(n_pts)
```

In [317]:
```
fig = plt.figure(dpi=200)
ax = fig.add_subplot(111)

ax.plot(ts[0], ys[0])
ax.plot(ts[1], ys[1])
ax.plot(ts[2], ys[2])
ax.plot(ts[3], ys[3])

Xs = np.linspace(0., t_final, num=20)
Ys = np.linspace(1., 20., num=20)

X, Y = np.meshgrid(Xs, Ys)
U = 1.
V = 1.5 * (Y - 2.)**(1. / 3.)
N = np.sqrt(U**2 + V**2)

U2, V2 = U / N, V / N
ax.quiver(X, Y, U2, V2)

plt.ylim([2., 10.])
plt.xlim([2., t_final])
plt.ylabel(r'$y(x)$')
plt.xlabel(r'$x$')
```

```
        plt.savefig("latex/files/prob_4.pdf")
        plt.show()
```

```
/Users/loganmorrison/.pyenv/versions/2.7.13/Python.framework/Versions/2.7/lib/python2.7/site-pa
```



1.4 Problem 5

Here our differential equation was

$$\frac{dy}{dx} + 2xy = xe^{-x^2} \tag{1}$$

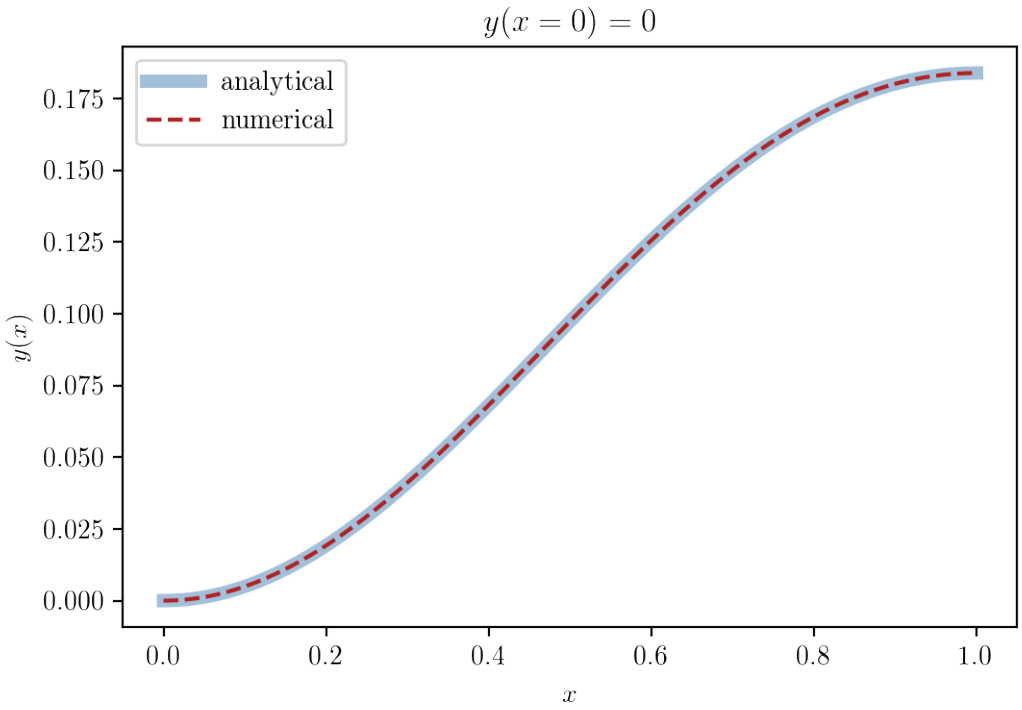
We found that the solutions were:

$$y(x) = \frac{1}{2}e^{-x^2}(x^2 + \mathcal{C}) \tag{2}$$

Let's check this by solving for the solutions using odeint from scipy. Let's check for $y(0) = 0$, which has $\mathcal{C} = 0$.

```
In [261]: def dydx(y, x):
              return - 2. * x * y + x * np.exp(-x**2)

          def a_soln(x):
              return 0.5 * x**2 * np.exp(-x**2)
```

```
xs = np.linspace(0.0, 1.0, num=150)
ys_numerical = odeint(dydx, 0.0, xs)
ys_analytical = np.vectorize(a_soln)(xs)

plt.figure(dpi=200)
plt.plot(xs, ys_analytical, label="analytical", lw=5, alpha=0.5)
plt.plot(xs, ys_numerical, ls='--', label="numerical",)
plt.legend()
plt.ylabel(r"$y(x)$")
plt.xlabel(r"$x$")
plt.title(r"$y(x=0) = 0$")
plt.savefig("latex/files/prob_5.pdf")
```



1.5 Problem Six

Here our differential equation was

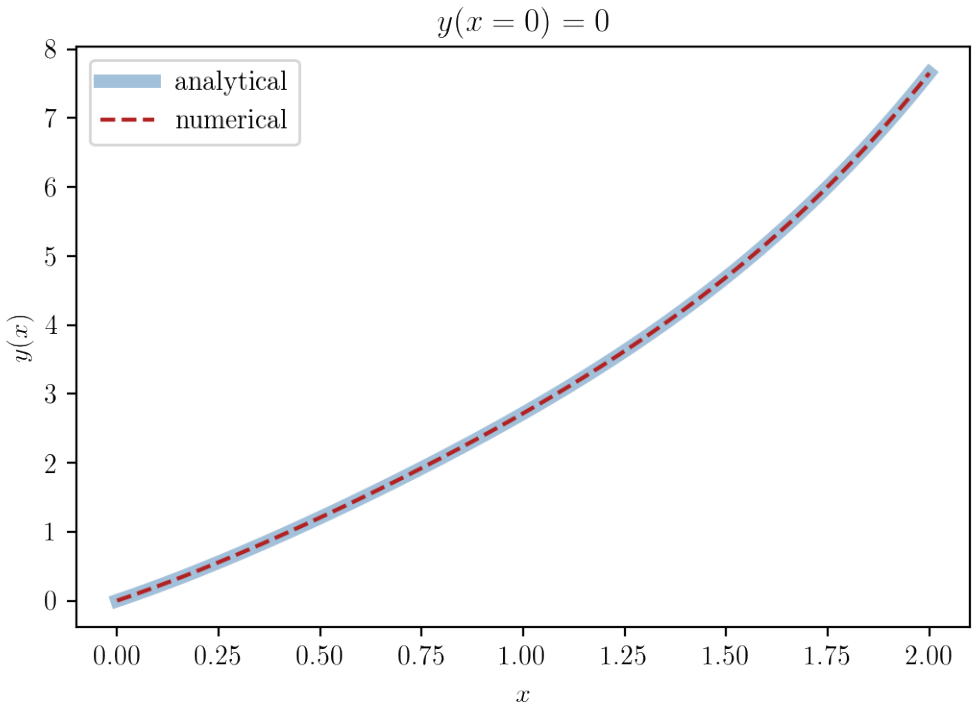
$$\frac{dy}{dx} + \cosh(x)y = 2e^x \tag{3}$$

We found that the solutions were:

$$y(x) = \frac{e^{2x} + 2x + \mathcal{C}}{\cosh(x)} \tag{4}$$

Let's check this by solving for the solutions using odeint from scipy. Let's check for $y(0) = 0$, which has $\mathcal{C} = -1$.

```
In [326]: def dydx(y, x):
              return 2. * np.exp(x) - np.tanh(x) * y

          def a_soln(x):
              return (np.exp(2. * x) + 2. * x - 1.0) / (2. * np.cosh(x))

          xs = np.linspace(0.0, 2.0, num=150)
          ys_numerical = odeint(dydx, 0.0, xs)
          ys_analytical = np.vectorize(a_soln)(xs)

          plt.figure(dpi=200)
          plt.plot(xs, ys_analytical, label="analytical", lw=5, alpha=0.5)
          plt.plot(xs, ys_numerical, ls='--', label="numerical",)
          plt.legend()
          plt.ylabel(r"$y(x)$")
          plt.xlabel(r"$x$")
          plt.title(r"$y(x=0) = 0$")
          plt.savefig("latex/files/prob_6.pdf")
```



1.6 Problem Eight

Here our differential equation was

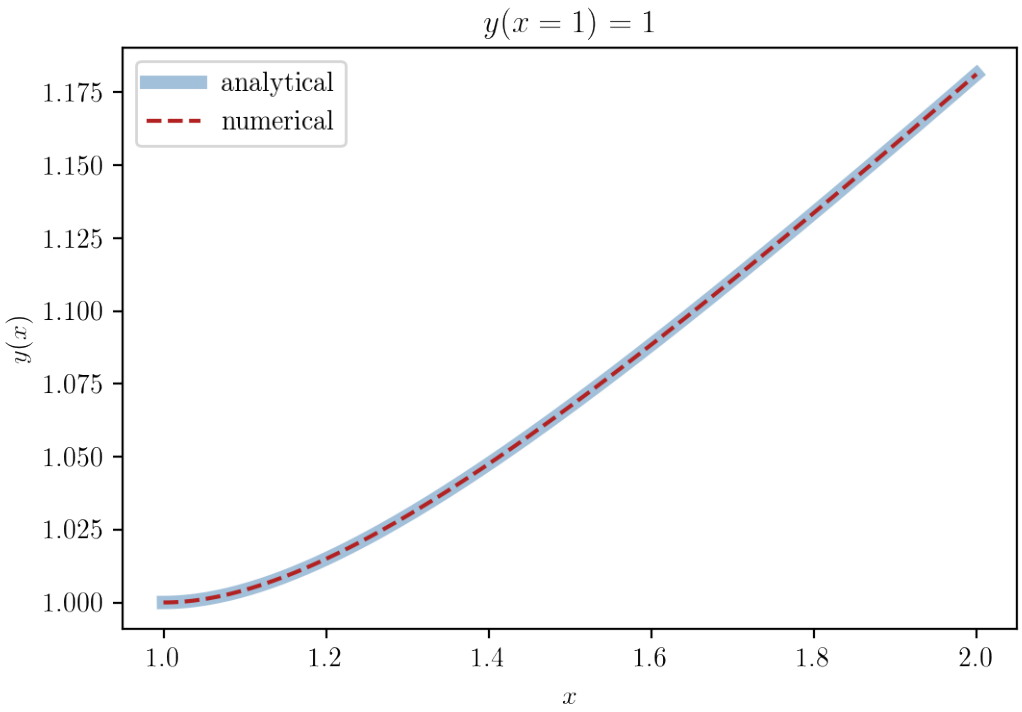
$$\frac{dy}{dx} = \frac{y}{x} - \left(\frac{y}{x}\right)^2 \tag{5}$$

We found that the solutions were:

$$y(x) = \frac{x}{\log(x) + C} \tag{6}$$

Let's check this by solving for the solutions using odeint from scipy. Let's check for $y(1) = 1$, which has $C = 1$.

```
In [325]: def dydx(y, x):
              return y / x - (y / x)**2

          def a_soln(x):
              return x / (np.log(x) + 1.)

          xs = np.linspace(1.0, 2.0, num=150)
          ys_numerical = odeint(dydx, 1.0, xs)
          ys_analytical = np.vectorize(a_soln)(xs)

          plt.figure(dpi=200)
          plt.plot(xs, ys_analytical, label="analytical", lw=5, alpha=0.5)
          plt.plot(xs, ys_numerical, ls='--', label="numerical",)
          plt.legend()
          plt.ylabel(r"$y(x)$")
          plt.xlabel(r"$x$")
          plt.title(r"$y(x=1) = 1$")
          plt.savefig("latex/files/prob_8.pdf")
```



1.7 Problem Nine

Here our differential equation was

$$\frac{dy}{dx} = -\frac{x}{y} + \sqrt{1 + \frac{x^2}{y^2}} \tag{7}$$

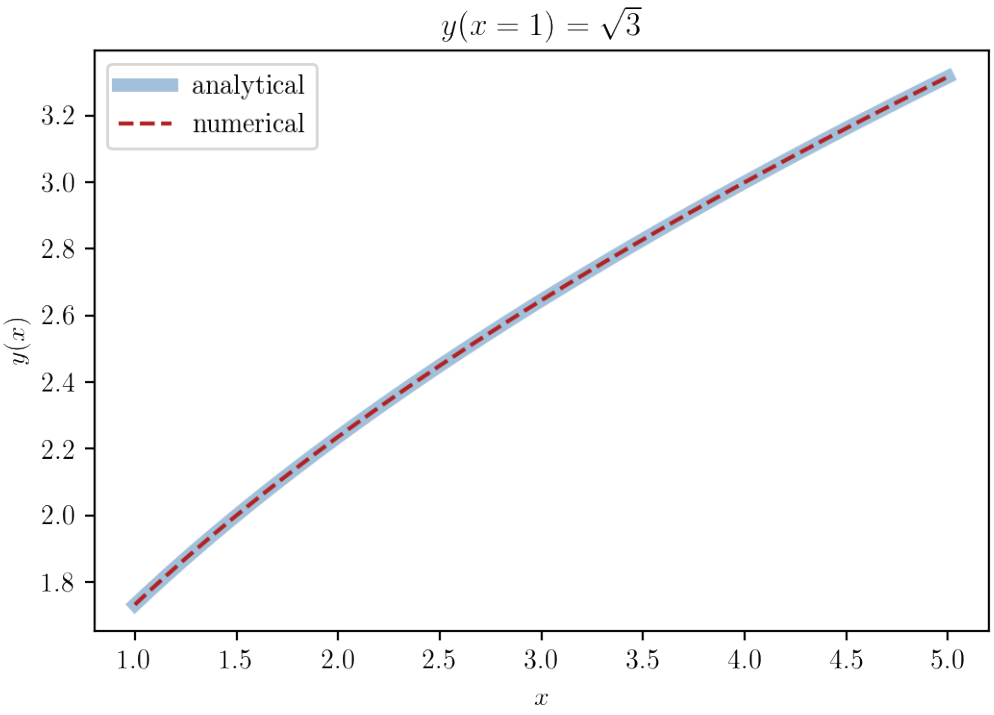
We found that the solutions were:

$$y(x) = \pm x\sqrt{(1 + \mathcal{C}/x)^2 - 1} \tag{8}$$

Let's check this by solving for the solutions using odeint from scipy. Let's check for $\mathcal{C} = 1$ and $y(1) = \sqrt{3}$

```
In [324]: def dydx(y, x):
              return -x / y + np.sqrt(1. + x**2 / y**2)

          def a_soln(x):
              return x * np.sqrt((1. + 1 / x)**2 - 1.)

          xs = np.linspace(1., 5.0, num=150)
          ys_numerical = odeint(dydx, np.sqrt(3), xs)
          ys_analytical = np.vectorize(a_soln)(xs)
```

```
plt.figure(dpi=200)
plt.plot(xs, ys_analytical, label="analytical", lw=5, alpha=0.5)
plt.plot(xs, ys_numerical, ls='--', label="numerical",)
plt.legend()
plt.ylabel(r"$y(x)$")
plt.xlabel(r"$x$")
plt.title(r"$y(x=1) = \sqrt{3}$")
plt.savefig("latex/files/prob_9.pdf")
```



1.8 Problem Ten

Here our differential equation was

$$\frac{dy}{dx} - 2y\cot(x) = \sin(x)\cos(x)/y \tag{9}$$

We found that the solutions were:

$$y(x) = \pm\sin^2(x)\sqrt{\mathcal{C} + \cot^2(x)/2} \tag{10}$$

Let's check this by solving for the solutions using odeint from scipy. Let's check for $\mathcal{C} = 1$ and $y(\pi/2) = 1$

```
In [323]: def dydx(y, x):
              return 2 * y / np.tan(x) + 0.5 * np.sin(2. * x) / y
```

```
def a_soln(x):
    return np.sqrt(np.sin(x)**4 - np.sin(2 * x)**2 / 4.)

xs = np.linspace(np.pi / 2., 3 * np.pi /4, num=150)
ys_numerical = odeint(dydx, 1., xs)
ys_analytical = np.vectorize(a_soln)(xs)

plt.figure(dpi=200)
plt.plot(xs, ys_analytical, label="analytical", lw=5, alpha=0.5)
plt.plot(xs, ys_numerical, ls='--', label="numerical",)
plt.legend()
plt.ylabel(r"$y(x)$")
plt.xlabel(r"$x$")
plt.title(r"$y(x=\pi/2) = 1$")
plt.savefig("latex/files/prob_10.pdf")
```