CS 7A - Spring 2015 - Project 1: Single Neuron Equation 2/19/15

Except for #1, write responses on separate paper.

1 Background Theory

Biologists are currently applying mathematical modeling to a wide range of problems. One such problem involves the function of nerve cells (neurons) in the brain. These cells generate electrical impulses that move along their axons, affecting the activity of neighboring cells. Even for a single neuron, a complete model using what is currently known bout it firing mechanism may take severalk equations, Since the human brain conatins on the order of 10^{10} neurons, an exact model is not feasible. In this application you will be looking at a very simplified model for an isolated population of neurons all having similar properties.

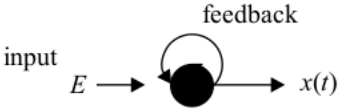


Figure 2.26. Input and output of the neuron population

Let $x(t)$ denote the percent of neurons firing at time t, normalized to be between 0 (low activity) and 1 (high activity). A simple model representing the change of activity level in the population is given by the differential equation

$$\frac{dx}{dt} = -x + S_a(x + E - \theta)$$

where E is the level of activity coming from the cells external to the population, θ is a common threshold level for cells in the set, and S_a is a response function that models the change in activity evel due to a given input. We will use a standard "sigmoidal" response function

$$S_a(z) = \frac{1}{1 + e^{-az}}$$

The nonlinear function S_a can be seen to increase monotonically from 0 to 1 as z increases from $infty$ to ∞. It is called a sigmoidal function because it has a sort of stylized S-shape. You may remember that solutions of the logistic growth equation had this same shape.

1. Find a formula for the derivative of $S_a(z)$, and show that it satisfies the identity

$$S'_a(z) \equiv aS_a(z)\,(1 - S_a(z))$$

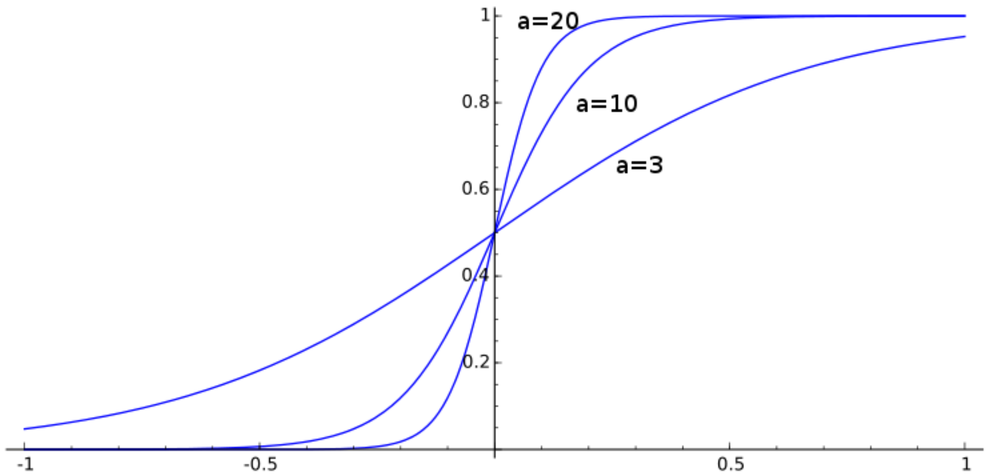
SOLN: $\frac{d}{dz}S_a(z) = \frac{d}{dz}(1+e^{-az})^{-1} = -(1+e^{-az})^{-2}(-ae^{-az}) = \dfrac{ae^{-az}}{(1+e^{-az})^2} = \dfrac{a}{1+e^{-az}} \cdot \dfrac{e^{-az}}{1+e^{-az}} =$ $aS_a(z)\left(\dfrac{1+e^{-az}}{1+e^{-az}} - \dfrac{1}{1+e^{-az}}\right) = aS_a(z)(1 - s_a(z))$

2. Draw a graph of $S_a(z)$ for $a = 3, 10$, and 20. Where is the slope a maximum? Is it the same in each case? Explain how the graph of $S_a(z - \theta)$ differs from the graph of $S_a(z)$.
SOLN: In sage, the commands

```
z,S=var('z S')
p0=plot(1/(1+e^(-3*z)),(z,-1,1))
p1=plot(1/(1+e^(-10*z)),(z,-1,1))
p2=plot(1/(1+e^(-20*z)),(z,-1,1))
show(p0+p1+p2)
```

produces the plot shown to the right. The slope is maximized for $z = 0$. $S_a(z-\theta)$ is a horizontal shift by θ. The model $\frac{dx}{dt} = -x + S_a(x + E - \theta)$, is autonomous and therefore it can be analyzed by drawing a phase line. Assume $a = 10$, and the incoming activity E is constant at $E = 0.2$ so that

$$\frac{dx}{dt} = -x + \frac{1}{1+e^{-10(x+0.2-\theta)}}$$



Because the value of S_a is always between 0 and 1, if $x > 1$ the slope $x' = S_a(x + 0.2 - \theta)$ is negative and if $x < 0$ it is positive. This means that any equilibrium solutions, that is, values of x where $\frac{dx}{dt} = 0$ must lie between 0 and 1. It also implies that the arrows on the phase line are always pointing down above $x = 1$ and up below $x = 0$.

Figure 2.27 shows the graphs of $y = x$ (the dashed line) and the response function $y = 1/(1 + e^{-10(x+0.2-\theta)})$ for $\theta = 0.4, 0.7.$ and 1.0. It can be seen that for small θ there will be one euqilibrium solution near $x = 1$ and for large θ there will be one equilibrium near $x = 0$. This seems reasonable since a high threshold means it takes a lot of input to produce very much activity. For θ in the middle range, however, there can be three equilibrium solutions.

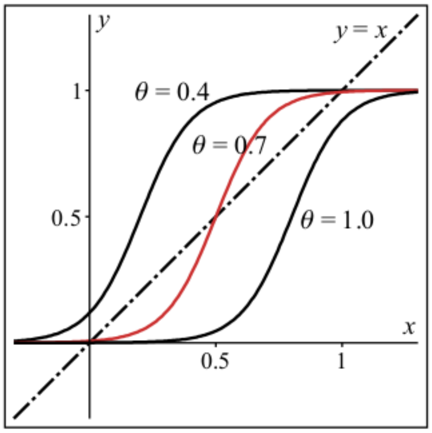


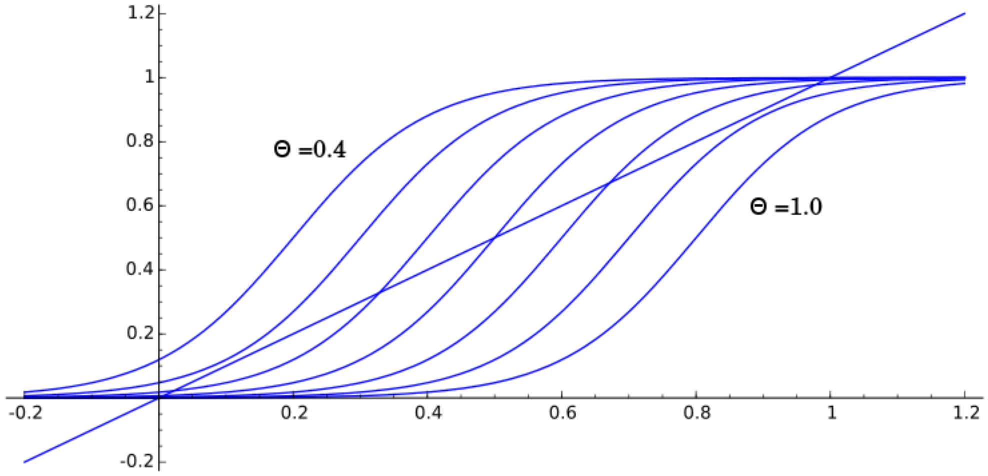
Figure 2.27. $y = 1/(1 + e^{-10(x+0.2-\theta)})$ for $\theta = 0.4, 0.7, 1.0$

3. Draw phase lines for $\frac{dx}{dt} = -x + \frac{1}{1+e^{-10(x+0.2-\theta)}}$ with $\theta = 0.4, 0.5, \ldots, 0.9, 1.0$ Label each equilibrium point as a sink, source or node. You will need a numerical equation solver to find the equilibrium values; that is, the

value of x where $x = \dfrac{1}{1+e^{-10(x+0.2-\theta)}}$. As a check, the three equilibria for $\theta = 0.7$ are $x_{\approx}0.007188, x_2 = 0.5$, and $x_3 \approx 0.992812$.
SOLN: To get some idea of where to hunt for what, it's helpful to have a graph of the seven curves for $\theta = 0.4, 0.5, \ldots, 0.9, 1.0$. In Sage,

```
g=Graphics()
x,y=var('x y')
p=plot(x,(x,-.2,1.2))
g=g+p
for i in range(4,11):
    p=plot(1/(1+e^(-10*(x+0.2-i/10))),(x,-0.2,1.2))
    g=g+p
show(g)
```

will produce

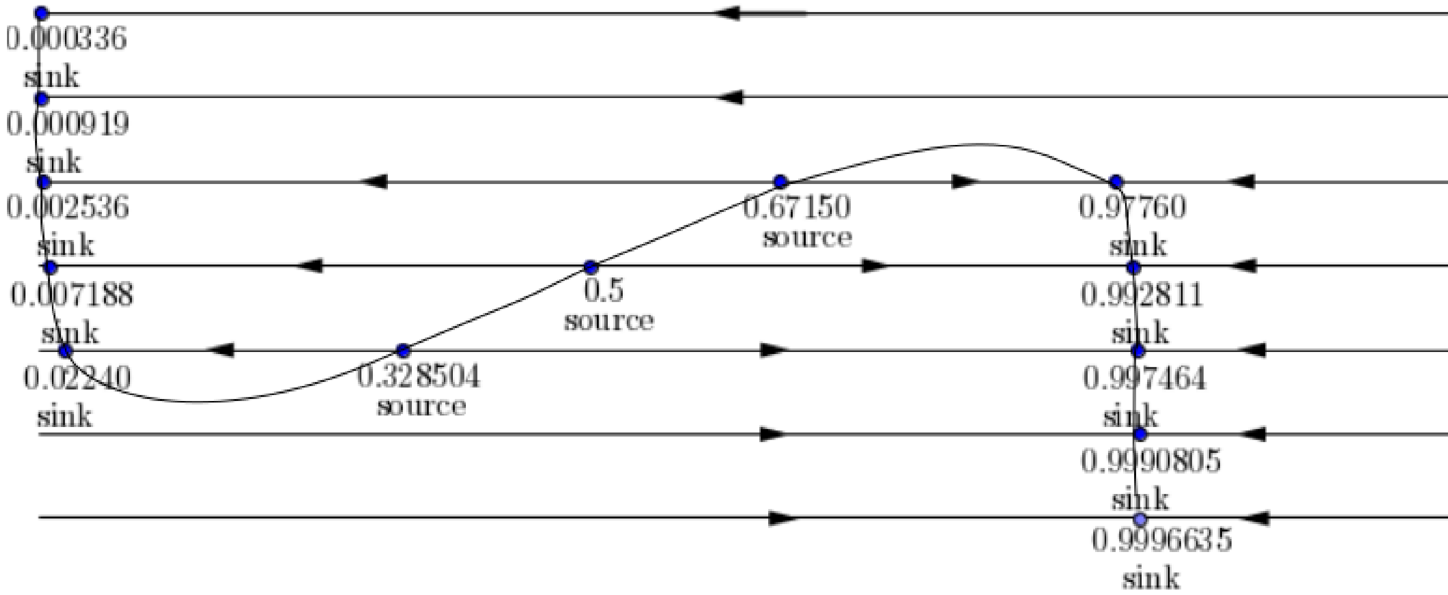


To find the numerical approximations in Sage, you can use the `find_root` command:

```
find_root(x==1/(1+e^(-10*(x+0.2-0.4))),0,1)
find_root(x==1/(1+e^(-10*(x+0.2-0.5))),0,1)
find_root(x==1/(1+e^(-10*(x+0.2-0.6))),0,1)
find_root(x==1/(1+e^(-10*(x+0.2-0.6))),0.2,0.4)
find_root(x==1/(1+e^(-10*(x+0.2-0.6))),0,0.1)
find_root(x==1/(1+e^(-10*(x+0.2-0.7))),0,1)
find_root(x==1/(1+e^(-10*(x+0.2-0.7))),0.4,0.6)
find_root(x==1/(1+e^(-10*(x+0.2-0.7))),0,0.1)
find_root(x==1/(1+e^(-10*(x+0.2-0.8))),0.9,1)
find_root(x==1/(1+e^(-10*(x+0.2-0.8))),0.6,0.7)
find_root(x==1/(1+e^(-10*(x+0.2-0.8))),0,0.1)
find_root(x==1/(1+e^(-10*(x+0.2-0.9))),0,1)
find_root(x==1/(1+e^(-10*(x+0.2-1))),0,1)
0.9996635199630014
0.9990805411739268
0.9974640310870945
0.3285041747454107
0.02240111117145591
0.9928119358173282
0.49999999999999983
0.007188064182671628
0.9775988888285436
```

```
0.6714958252545886
0.0025359689129061396
0.0009194588260732535
0.0003364800369985759
```

With these values and sampling the sign of $x'(t)$ in the various intervals, the phase lines shown below were constructed using Geogebra and Gimp:



4. Make a bifurcation diagram by putting the seven phase lines from problem (3) in a row and joining the equilibrium points.
   SOLN: The easiest way to get the bifurcation curve is just to connect the dots by hand (or in your mind) with a smooth curve that has a local min somewhere between the lines $\theta = 0.5$ and $\theta = 0.6$ and a local max somewhere between $\theta = o.7$ and $\theta = 0.8$ and has the sigmoid shape of an odd powered polynomial. I used Sage to fit a fifth degree polynomial to the equilibrium points from the phase lines and the results are shown below:

```
data=[(0.9996635199630014,0.4),(0.9990805411739268,0.5),(0.9974640310870945,.6),
(0.3285041747454107,.6),(0.02240111117145591,.6),(0.9928119358173282,.7),(.5,.7),
(0.007188064182671628,.7),(0.9775988888285436,.8),(0.6714958252545886,.8),
(0.0025359689129061396,.8),(0.0009194588260732535,.9),(0.0003364800369985759,1)]
var('a, b, c, d, e, f, x')
model(x) = a*x^5+b*x^4+c*x^3+d*x^2+e*x+f
find_fit(data, model)
```

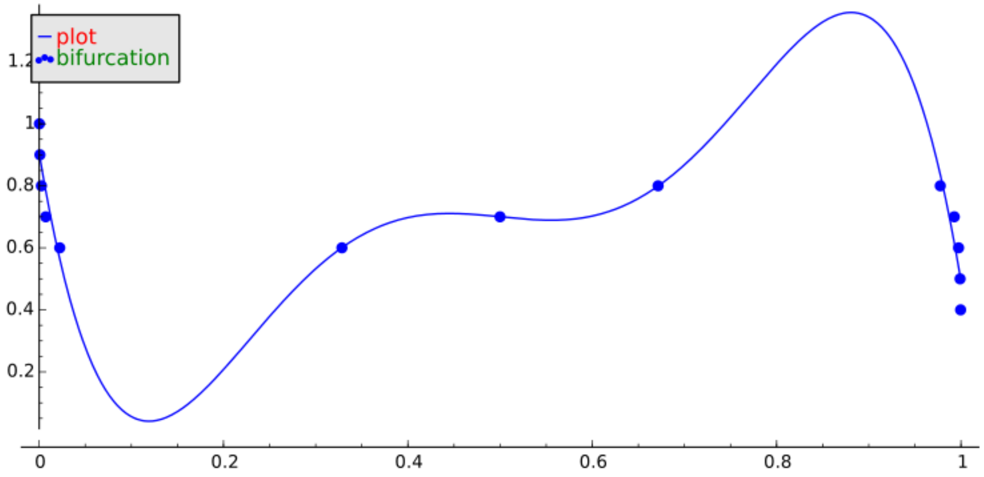
This produced these coefficients:

```
[a == -137.7116256950839, b == 344.2790465872066, c == -310.2946819311529,
d == 121.16298975973874, e == -17.84809646219514, f == 0.9061838446773811]
```

I then created a plot of these like so:

```
P = points(data, pointsize=40, legend_label='bifurcation', legend_color='red')
P + plot(-137.711625695*x^5+344.279046587*x^4-310.2946819312*x^3+121.16298976*x^2
-17.8480964622*x+0.9061838447,(x,0,1), legend_label='plot', legend_color='green')
show(P)
```

But the result is not satisfying since the local min and local max are obviously way off:



5. From the bifurcation diagram, estimate the two bifurcation values of θ where the number of equilibrium points changes from one to three, and then from three back to one. SOLN: I think it's better to use the figure showing the functions $1/(1+e^{-10(x+0.2-\theta)})$ superimposed on $y=x$ from which I surmise that the values are about $\theta = 0.52$ and $\theta = 0.88$

6. Find the two bifurcation values of θ analytically. You will need to solve two simultaneous equations obtained by using the fact that at a bifurcation value of θ the curves $y = 1/(1+e^{-10(x+0.2-\theta)})$ and $y=x$ have a point of tangency. At this point the y-values are equal and the slopes are also equal.
   SOLN: We want to solve the system of equations

$$
\begin{aligned}
x &= \frac{1}{1+e^{-10(x+0.2-\theta)}} = S_{10}(x-0.2-\theta) \\
\frac{dx}{dt} &= 1 = S'(x+0.2-\theta) = 10S_{10}(x+0.2-\theta)(1-S_{10}(x+0.2-\theta))
\end{aligned}
$$

Combining these equations we can eliminate θ for $1 = 10x(1-x)$ or $10x^2 - 10x - 1 = 0 \Leftrightarrow x = \dfrac{5 \pm \sqrt{15}}{10}$
We can now substitute back and solve for θ. In Sage, this can be done with

```
var('theta')
x=1/10*sqrt(15) + 1/2
show(solve(x==1/(1+e^(-10*(x+0.2-theta))),theta));
n(1/10*sqrt(15) + 1/10*log(-sqrt(15)/(sqrt(15) + 5) + 5/(sqrt(15) + 5)) + 7/10);show(theta)
```

or $\theta = \frac{1}{10}\sqrt{15} + \frac{1}{10}\log\left(-\frac{\sqrt{15}}{\sqrt{15}+5} + \frac{5}{\sqrt{15}+5}\right) + \frac{7}{10} \approx 0.8809546$

Without the insight that we can get a quadratic in x, teasing a numerical solution to a system of non-linear equations out of Sage is not straightforward. The system we want to solve is

$$
\begin{aligned}
x &= 1/(1+e^{-10(x+0.2-\theta)}) \qquad (1) \\
1 &= 10e^{10\theta-10x-2}/(e^{10\theta-10x-2}+1)^2, \qquad (2)
\end{aligned}
$$

However, these commands and the response show no satisfaction:

```
t,x=var('t x')
solve([x==1/(1+e^(-10*(x+0.2-t))),1==10*e^(10*t - 10*x - 2)/(e^(10*t - 10*x - 2) + 1)^2],t,x)
[1 == 10*e^(10*t - 10*x - 2)/(e^(10*t - 10*x - 2) + 1)^2, x == (1/(e^(10*t - 10*x - 2.0) + 1))]
```

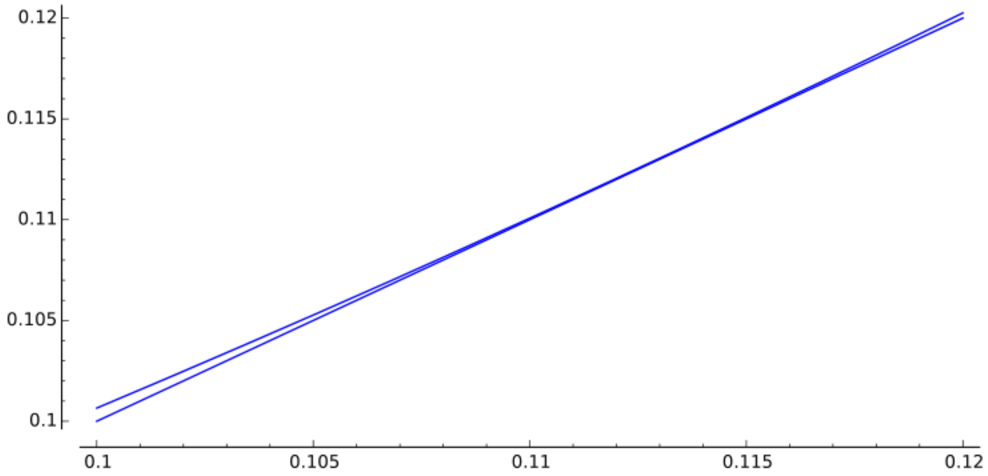
So I did a bit a research and tried this:

```
t,x=var('t x')
eq1=x==1/(1+e^(-10*(x+0.2-t)))
eq2=10*e^(10*t - 10*x - 2)/(e^(10*t - 10*x - 2) + 1)^2==1
solns = solve([eq1,eq2],t,x,solution_dict=True)
[[s[t].n(30), s[x].n(30)] for s in solns]
```

But, alas, this also just spit back the original system, now with a "Key Error" for the last line.
Wolfram Alfa chugs away at it for a while and then gives up.
So we resort to guess and check. We already know about where to look, after all.

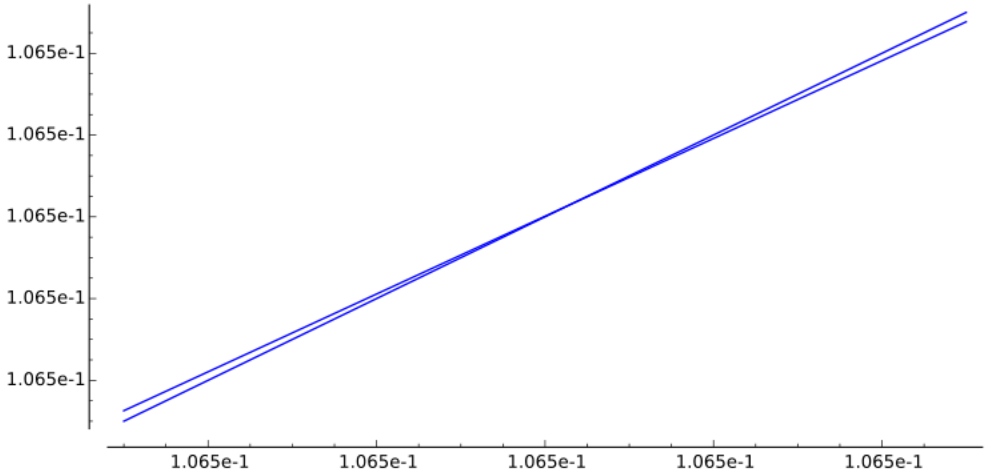
```
g=Graphics()
x,y=var('x y')
p=plot(x,(x,.1,.12))
g=g+p
p=plot(1/(1+e^(-10*(x+0.2-.519))),(x,0.1,.12))
g=g+p
show(g)
```

Here we have $\theta = 0.519$ and we get this graph for $0.1 \leq x \leq 0.12$:



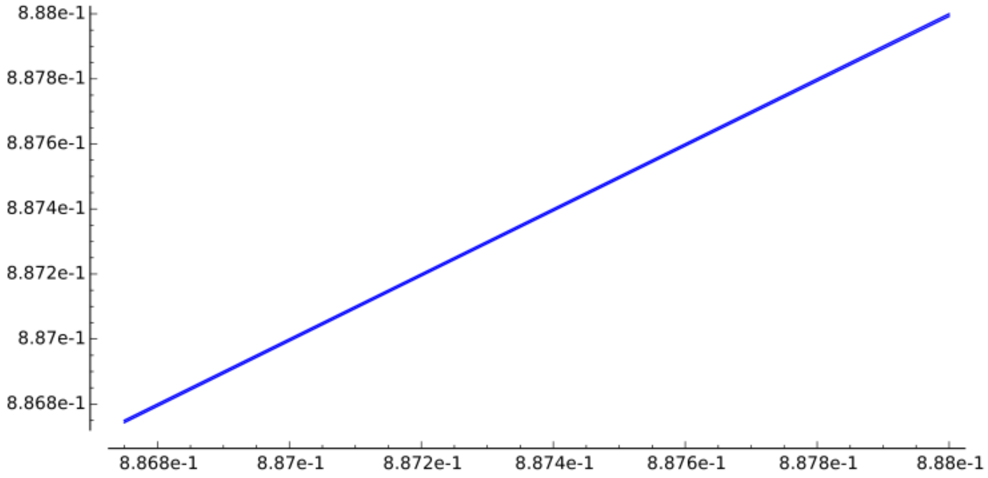
Zooming in through more trial and error:

```
g=Graphics()
x,y=var('x y')
p=plot(x,(x,.10645,.10655))
g=g+p
p=plot(1/(1+e^(-10*(x+0.2-.5192))),(x,0.10645,.10655))
g=g+p
show(g)
```



This suggests that $\theta \approx 0.5192, x \approx 0.1065$.
At the other end we can do the same sort of zooming leading to something like this:

```
g=Graphics()
x,y=var('x y')
p=plot(x,(x,.88675,.888))
g=g+p
p=plot(1/(1+e^(-10*(x+0.2-.88096))),(x,0.88675,.888))
g=g+p
show(g)
```



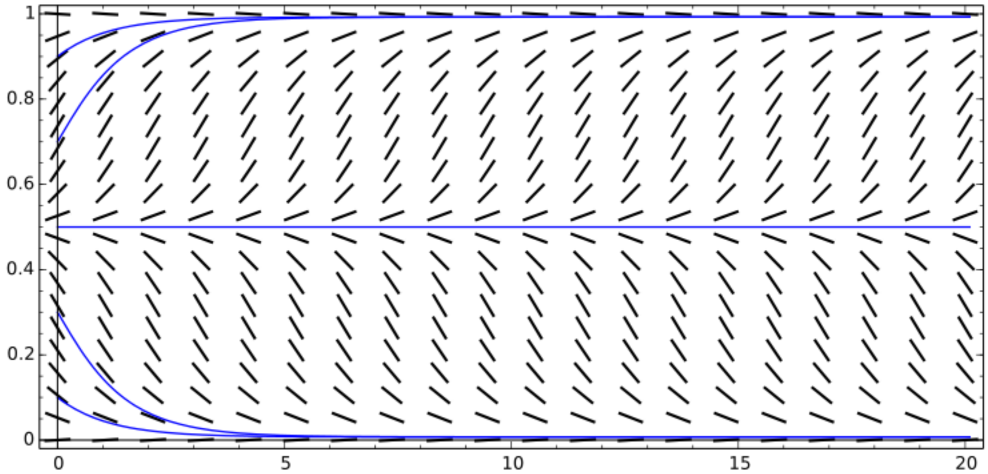
Showing that $\theta \approx 0.8810, x \approx 0.8873$
It was only when I heard Tuan say, ”You can solve it analytically!” that I turned my attention to that possibility! Oh well...laziness advances with age, perhaps...but encouraged by Tuan's confidence, I looked back at the situation and realized that since, as we established in problem #1, $S_a'(z) \equiv aS_a(z)\,(1 - S_a(z))$, and we seek to solve the system that says $x = S_a(x)$ and $S_a'(x) = 1$ which means that $x(1-x) = 1 \Leftrightarrow x^2 - x = -1$

7. Use your computer algebra system to draw a slope field for the ode with $a = 10, E = 0.2$, and $\theta = 0.7$. Let t vary from 0 to 20. Use initial values $x(0) = 0.1, 0.3, 0.5, 0.7$, and 0.9, and describe what happens to the activity as $t \to \infty$.
SOLN: In order to accomplish this in Sage, it's helpful to write the code for the building the numerical solutions. The code below was adapted from the Mendel University (located in Brno, Czech Republic) site http://user.mendelu.cz/marik/sage/dr.pdf. In Czeck, “krok” must mean something like “step size”:

```
t,x=var('t x')
```

```
def rk(fnc,t0,x0,krok,t1):
    g=fast_float(fnc,'t x','x')
    n=int((1.0)*(t1-t0)/krok)
    t00=t0; x00=x0
    soln=[[t00,x00]]
    for i in range(n+1):
        m1=g(t00,x00)
        m2=g(t00+krok/2,x00+m1*krok/2)
        x00=x00+krok*m2
        t00=t00+krok
        soln.append([t00,x00])
    return soln

f(t,x)=-x+1/(1+e^(-10*(x+0.2-0.7)))
A=plot_slope_field(f(t,x),(t, t_min, t_max), (x, x_min, x_max))
B=list_plot(rk(f(t,x), 0, 0.1, 0.1, t_max), plotjoined=True)
C=list_plot(rk(f(t,x), 0, 0.3, 0.1, t_max), plotjoined=True)
D=list_plot(rk(f(t,x), 0, 0.5, 0.1, t_max), plotjoined=True)
E=list_plot(rk(f(t,x), 0, 0.7, 0.1, t_max), plotjoined=True)
F=list_plot(rk(f(t,x), 0, 0.9, 0.1, t_max), plotjoined=True)
show(A+B+C+D+E+F,ymin=x_min,ymax=x_max,xmin=t_min,xmax=t_max)
```

This is somewhat shy of `rk4`, but it seems to work well enough and we get the slope field `A` on which we can use `list_plot` to superimpose the 5 solutions.



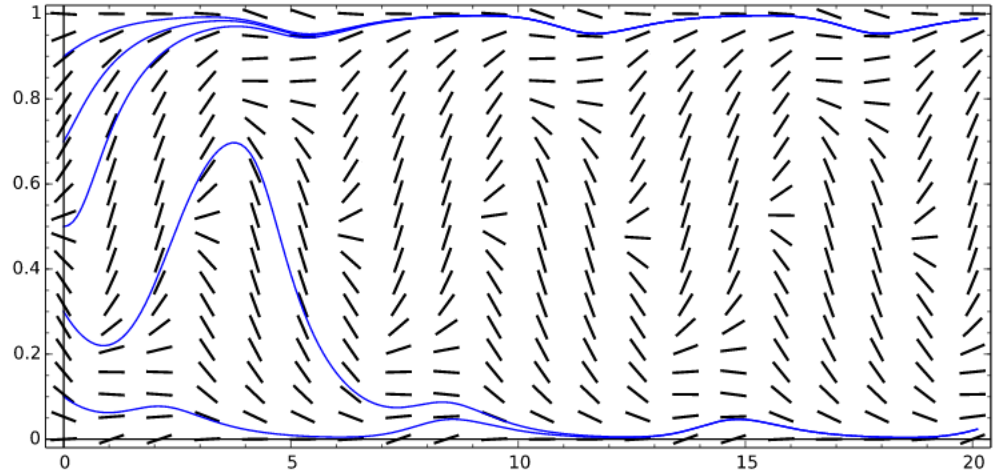
Evidently, solutions with initial values above $x(0) = 0.5$ tend towards the equilibrium $x \approx 0.8873$ and initial values below $x_0 = 0.5$ tend towards the sink at $x \approx 0.1065$. If you start dead on the unstable equilibrium at $x = 0.5$, however, "you" will stay on it.

8. Redo problem (7) with periodic input $E = E(t) = 0.2(1 + \sin(t))$. With this time varying input the equation is no longer autonomous. Explain carefully how the activity differs from that described in (7). Do you think there is a periodic solution separating the two types of solutions in the periodic case? This is an interesting problem and it might be a good time to look at the paper "*Qualitative tools for studying periodic solutions and bifurcations as applied to the periodically harvested logistic equation*" by Diego Benardete, V.W.Noonburg, and B. Pollina, Amer. Math. Monthy, vol 115, 202-219(2008). It discusses, in a very readable way, how one goes about determining the answer to such a question.
SOLN: At first it seemed it would be simple enough to replace the 0.2 in the exponent of x' with the sinusoid in Sage, but as you may have come to expect, you get a cascade of red error messages. This caused me to look at the help file for the `fast_flot()` function and modify it's call as shown below:

```
t,x=var('t x')
```

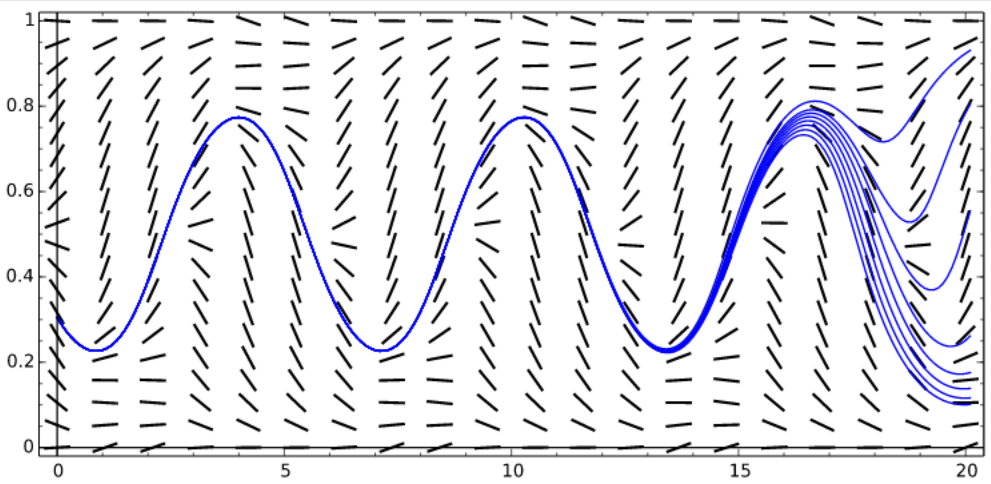
```
(t_min, t_max, x_min,x_max)=(0,20,0,1)
def rk(fnc,t0,x0,krok,t1):
    g=fast_float(fnc,’t’,’x’) #,’x’)
    n=int((1.0)*(t1-t0)/krok)
    t00=t0; x00=x0
    soln=[[t00,x00]]
    for i in range(n+1):
        m1=g(t00,x00)
        m2=g(t00+krok/2,x00+m1*krok/2)
        x00=x00+krok*m2
        t00=t00+krok
        soln.append([t00,x00])
    return soln

f(t,x)=-x+1/(1+e^(-10*(x+0.2*(1+sin(t))-0.7)))
A=plot_slope_field(f(t,x),(t, t_min, t_max), (x, x_min, x_max))
B=list_plot(rk(f(t,x), 0, 0.1, 0.1, t_max), plotjoined=True)
C=list_plot(rk(f(t,x), 0, 0.3, 0.1, t_max), plotjoined=True)
D=list_plot(rk(f(t,x), 0, 0.5, 0.1, t_max), plotjoined=True)
E=list_plot(rk(f(t,x), 0, 0.7, 0.1, t_max), plotjoined=True)
F=list_plot(rk(f(t,x), 0, 0.9, 0.1, t_max), plotjoined=True)
show(A+B+C+D+E+F,ymin=x_min,ymax=x_max,xmin=t_min,xmax=t_max)
```

This worked!



Yes, I think there is an initial condition x_0 which will follow the unstable oscillatory equilibrium in the middle, but no numerical approximation for the solution exists because an deviation, no matter how slight, from the true solution will cause it to diverge towards one of the stable solutions. To support this, I did some experiments which suggest the initial value in question is in the neighborhood of $x_0 \approx 0.304668$ as suggested by the Sage code and graph below:

```
B=list_plot(rk(f(t,x), 0, 0.3046687, 0.1, t_max), plotjoined=True)
C=list_plot(rk(f(t,x), 0, 0.3046686, 0.1, t_max), plotjoined=True)
D=list_plot(rk(f(t,x), 0, 0.3046685, 0.1, t_max), plotjoined=True)
E=list_plot(rk(f(t,x), 0, 0.3046684, 0.1, t_max), plotjoined=True)
F=list_plot(rk(f(t,x), 0, 0.3046683, 0.1, t_max), plotjoined=True)
G=list_plot(rk(f(t,x), 0, 0.3046682, 0.1, t_max), plotjoined=True)
H=list_plot(rk(f(t,x), 0, 0.3046681, 0.1, t_max), plotjoined=True)
I=list_plot(rk(f(t,x), 0, 0.304669, 0.1, t_max), plotjoined=True)
show(A+B+C+D+E+F+G+H+I,ymin=x_min,ymax=x_max,xmin=t_min,xmax=t_max)
```


1
0.8
0.6
0.4
0.2
0
0
5
10
15
20