**1vishnukeyenDate - 14/08/2015**

1. Initialised git repo at hamborger/brian.git
2. uploaded stable.py,unstable.py and stateSpace.py to github
3. Made a backup in the /storage under himanshu/brian
4. BRIAN version 1.3.1 is being used.
5. Explanation on how sigma for a group is determined:  
     
   In stateSpace.py, function estimate\_params is used to get sigma. The estimate\_params function takes in two input arguments, mon ‘spike monitor’ and time\_est ‘initial burst time’ specified in default\_params structure. The initial\_burst\_t is 50msec. In estimate\_params the spike times from the last layer are taken as input. Zip(\*mon.spikes) puts the spike time lists from all the 100 neurons in one package. The spike times, which is 100 x runtime(90ms, step size 0.1), in a time window of 30ms + synaptic delay from the initial burst time are taken. A standard deviation of spike times is then calculated in that time window by finding the mean spike time.   
     
   In short the logic is to determine the variation in spike times developed from the case if everything is synchronous in that time window.

Code snippets:

(91) initial\_burst\_t=50 \* ms

(196) net.run() #Runs the network simulation

(153) i, times = zip(\*mon.spikes)

(154) times = array(times) #datatype conversion from simple list to array

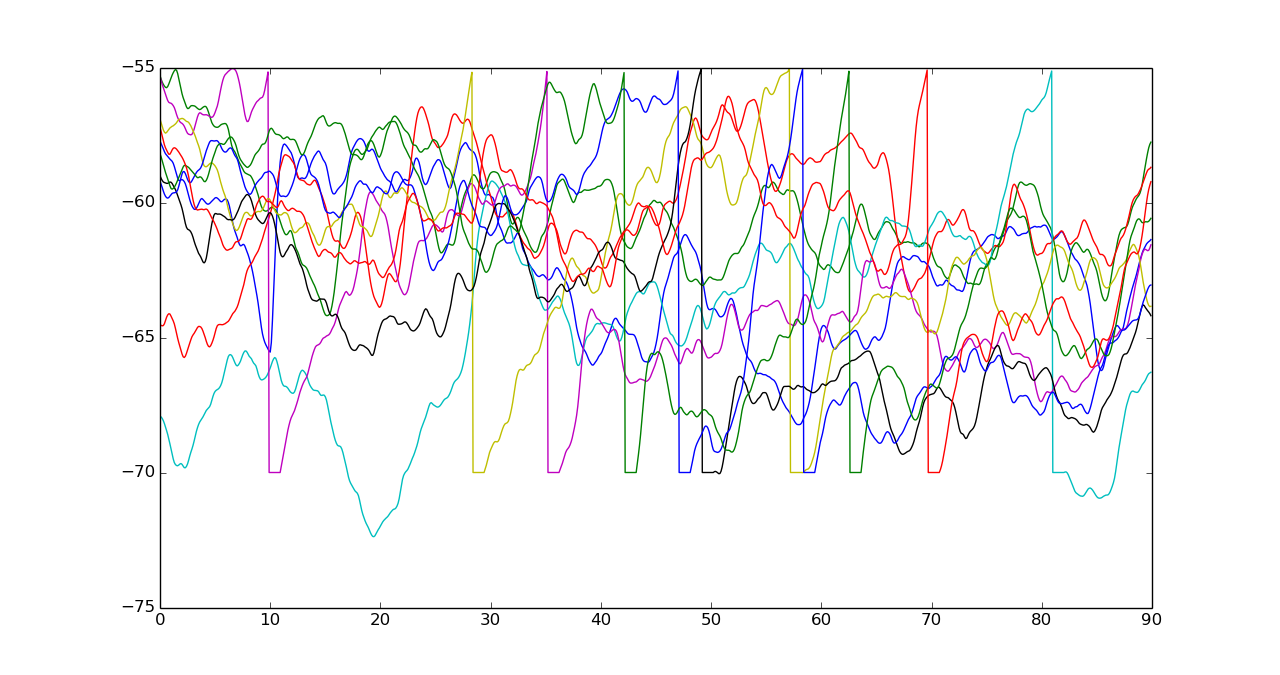
(155) times = times[abs(times - time\_est) < 15 \* ms]

(162) return (len(times),times.std())

(197) (newa, newsigma) = estimate\_params(net.mon[-1], params.initial\_burst\_t) #newsigma is the standard deviation for the last layer.

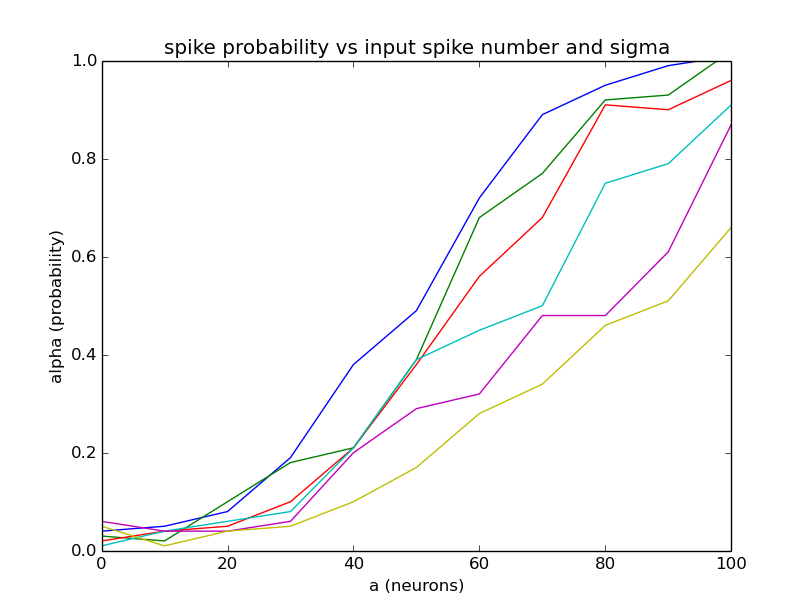
**Date - 17/08/2015**

1. Made changes in the stateSpace.py default\_params initial\_burst\_t set to 10ms not 50ms.
2. In stable.py added user input for a,sigma
3. StateMonitor(P,’v’,record=[1,101,201,301,401,501,601,701,801,901])   
   Instead of SpikeMonitor this function will give the voltage trace for the selected neurons out of total 1000 neurons.  
     
   X axis : runtime(in msec) ,Y axis : Membrane voltage(in mV)  
   Ymax : threshold voltage  
   Ymin : resting potential

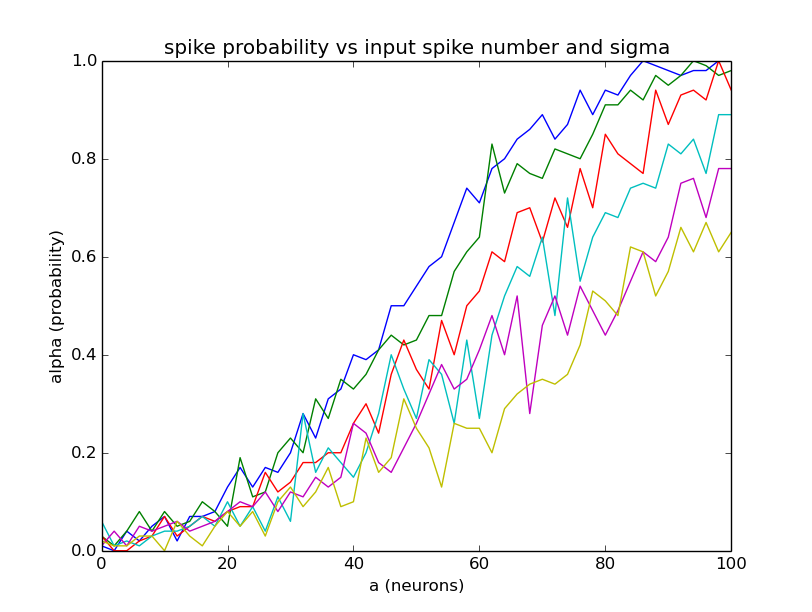


4. I reproduced figure 2C from Diesmann et. al. 1999 paper. In stateSpace.py, sigma\_vs\_a function calculates the probability or percentage of spikes occurring in the last group by finding new a and dividing it by the number of neurons in that layer which is 100. It does so by changing the input volley’s neuronal number ‘a’ fixing sigma. Hence we get response probabilities accordingly for different temporal dispersion in input.

In the figure below it is evident that violet (sigma=0ms) has the highest probability of response compared to yellow (sigma=5ms) where each line is evaluated for 10 different values of a.



Plotted for 50 equally spaced a’s from 0 to 100.



**Date - 18/08/2015**

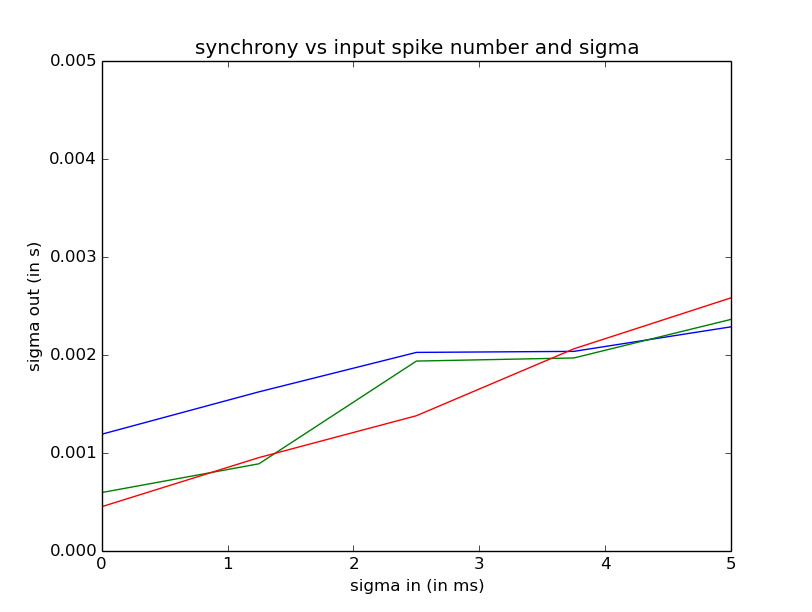
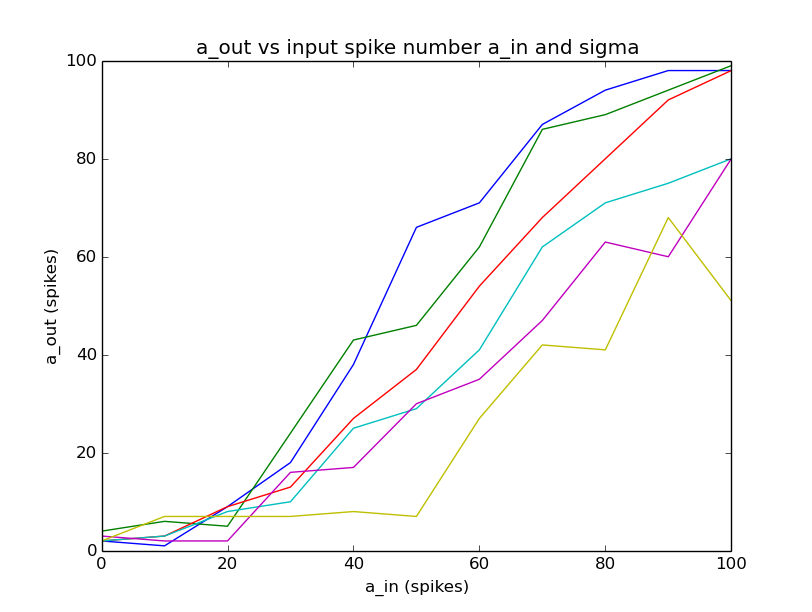
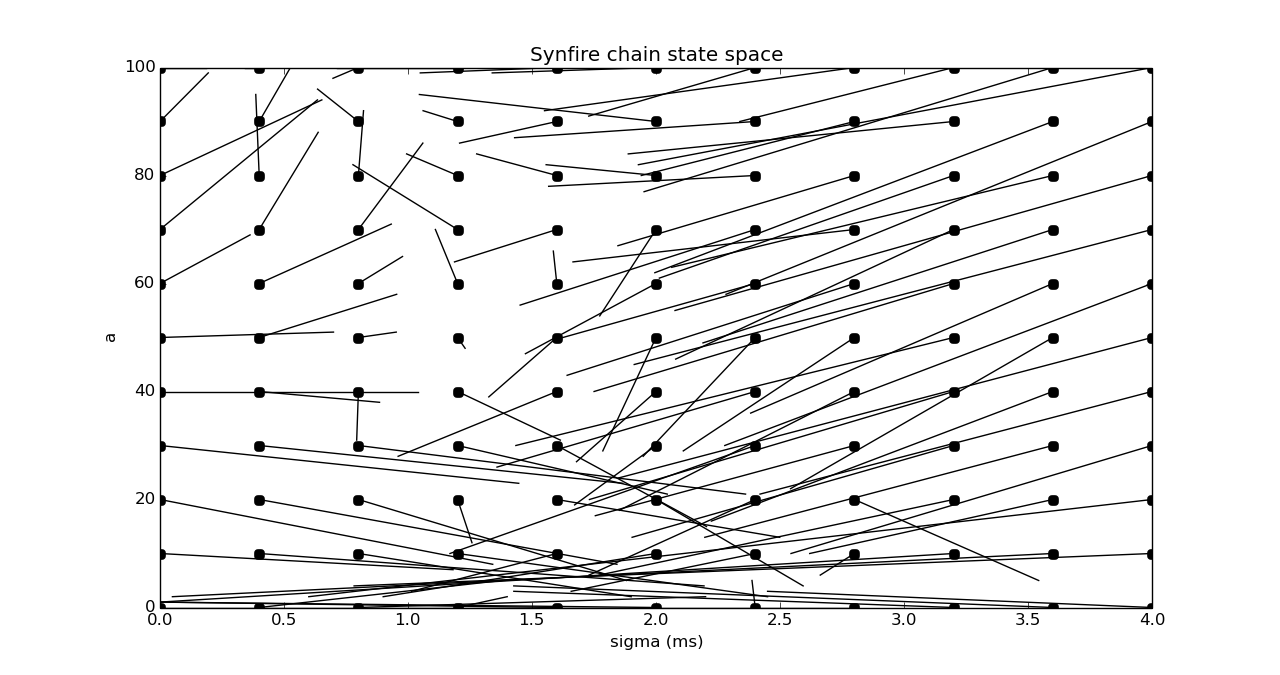
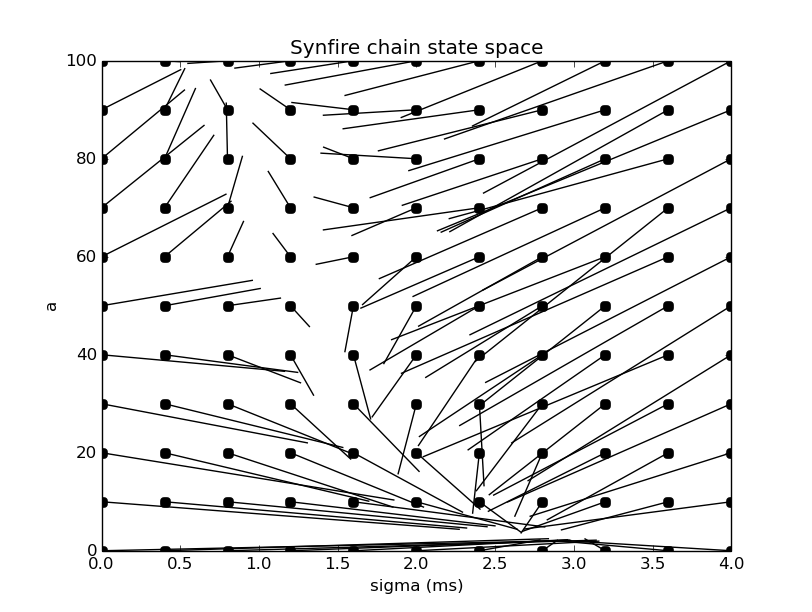
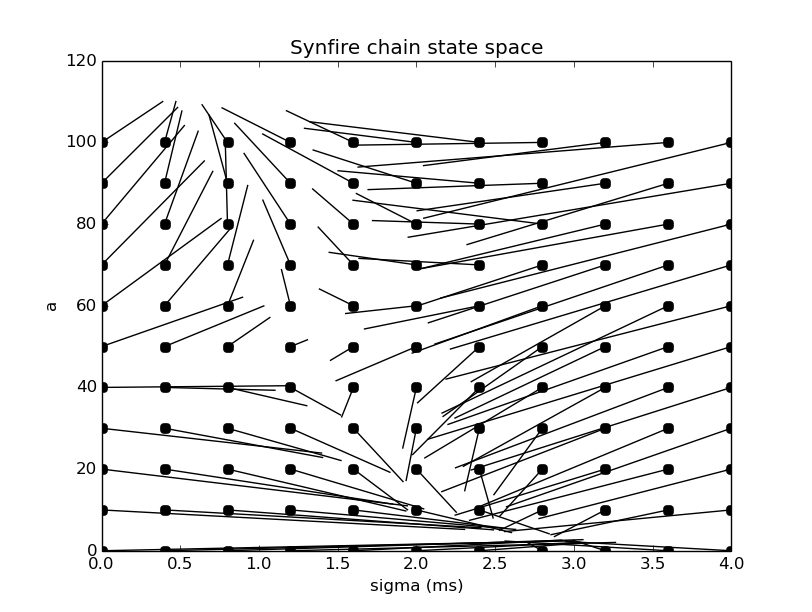
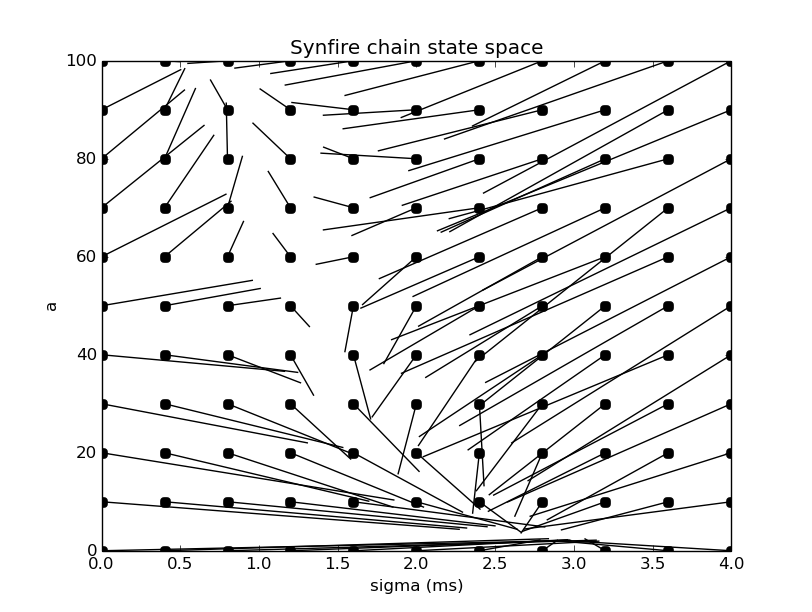
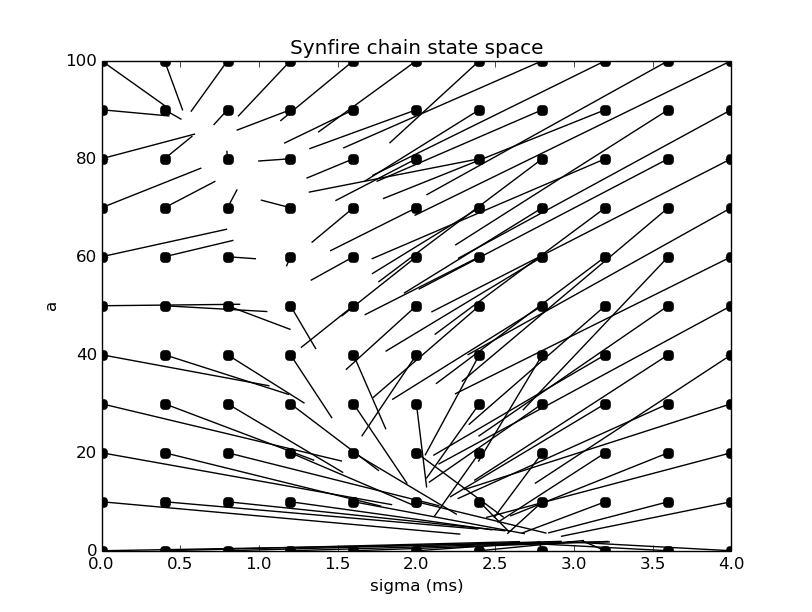
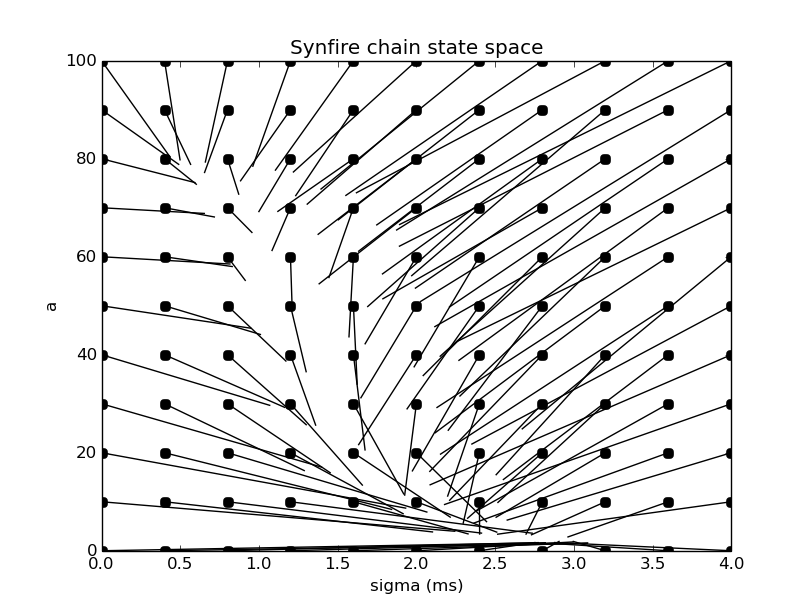
1. Plotted sigma out vs sigma in for different numbers of neurons ‘a’. It is seen that sigma is not dependent on ‘a’ where blue is 45, green 65 and red 100.
2. To reproduce figure 3a, figure 4(a,b,c,d)
3. Figure 3a:  
   In this figure the output of spikes from the last group is measured with respect to input spikes having fixed temporal dispersion of 0(Blue)-5ms(Yellow). This plot has lines each containing 10 points.  
   

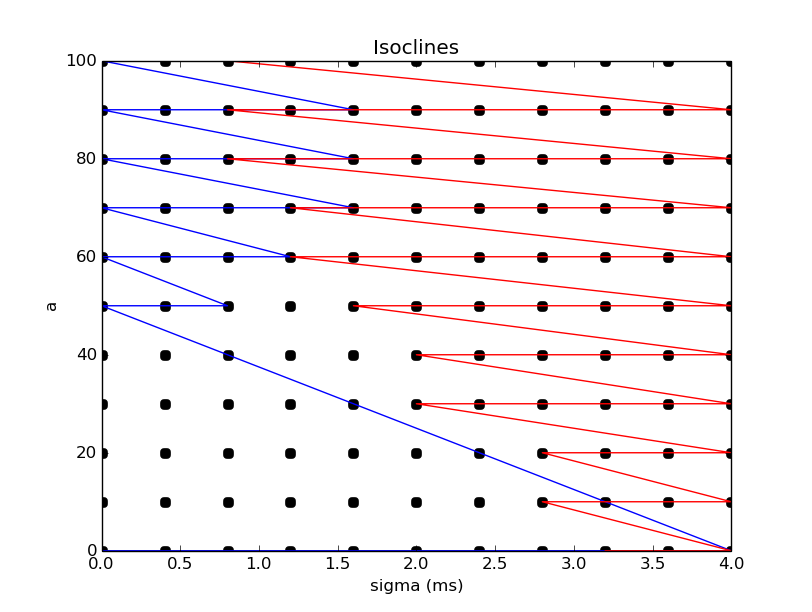
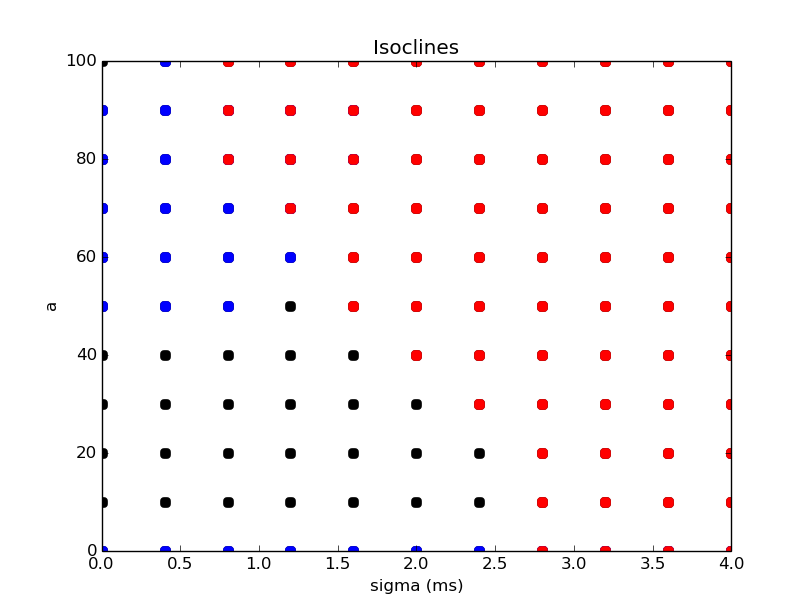
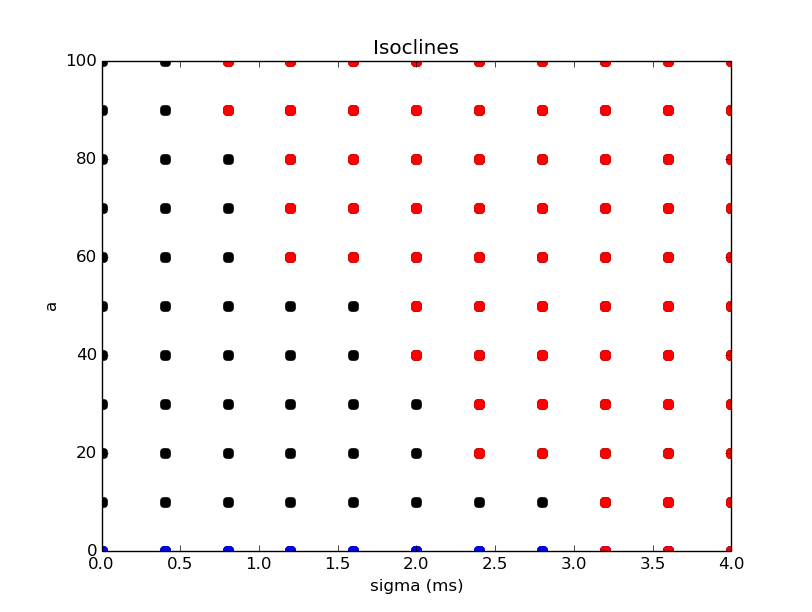
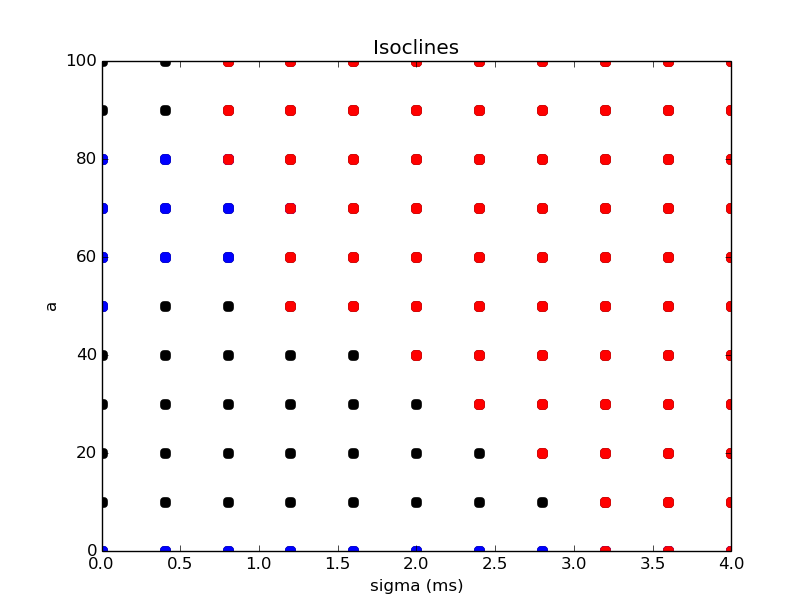
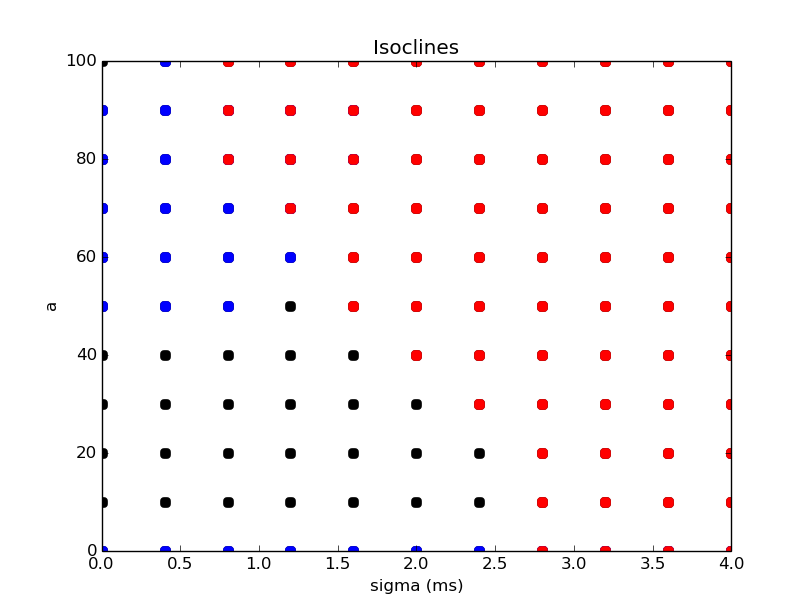
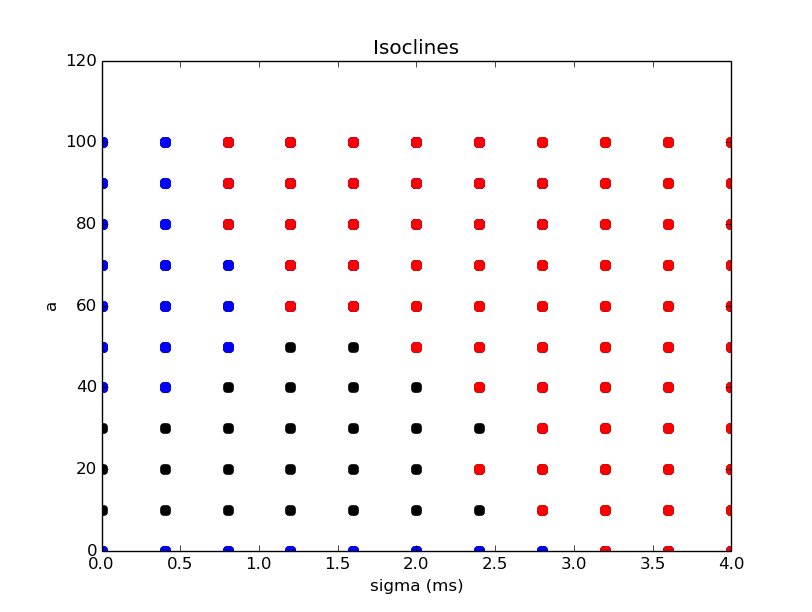
Figure 3c:  
State space plot generated using stateSpace.py with a grid of 10x10 size in the alpha-sigma phase plane. In the state space plot the top left region is converging to the stable fixed point defined as propagation with increasing synchronisation. The central region is the saddle node where the propagation can be either stable or unstable defined as propagation with increasing spread or dispersion in spike times. Neuron\_multiplicity is 1 and 50 for the corresponding figures below. Neuron\_multiplicity increases the probability of stable propagation by changing the number of neurons in a layer by a factor. Then new a is calculated by dividing the newa, returned by estimate\_param function, by neuronal\_muliplicity scaling it back to 100 or w as set in the model\_parameters. A neuron\_multiplicity of 50 means that there are 5000 (100 x 50) neurons per layer. It is observed that as we increase the multiplicity number the central region of the state space corresponding to a saddle node shrinks.

1. 
2. Figure 4a,4b,4c,4d:  
   In the plots below the role of neurons per layer w is being investigated. It is seen that as w is increased from 80 to 90 a stable fixed point comes into the picture. The stable fixed point shifts top-leftwards as we further increase the w from 90 to 110. 

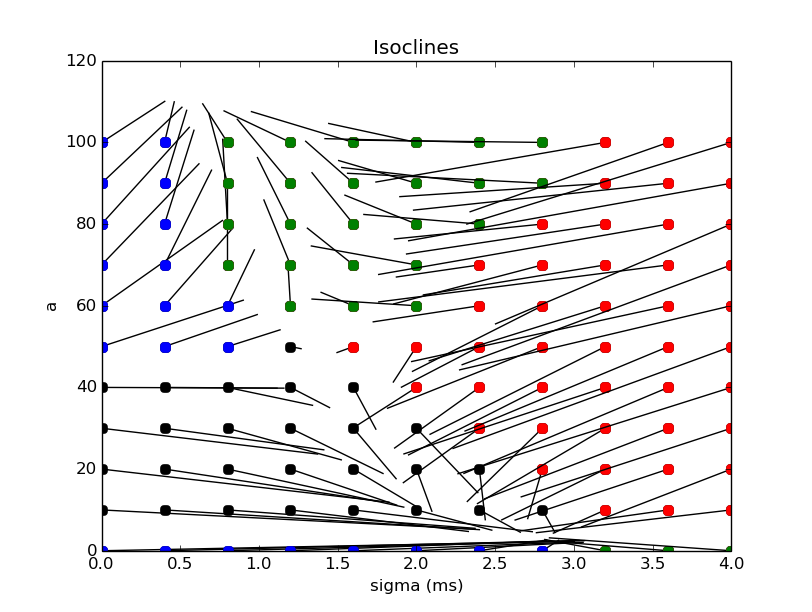
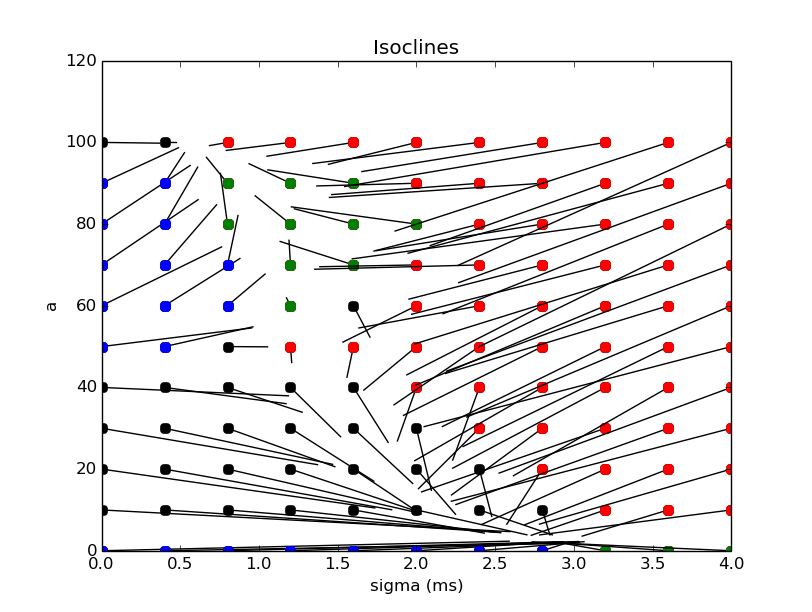
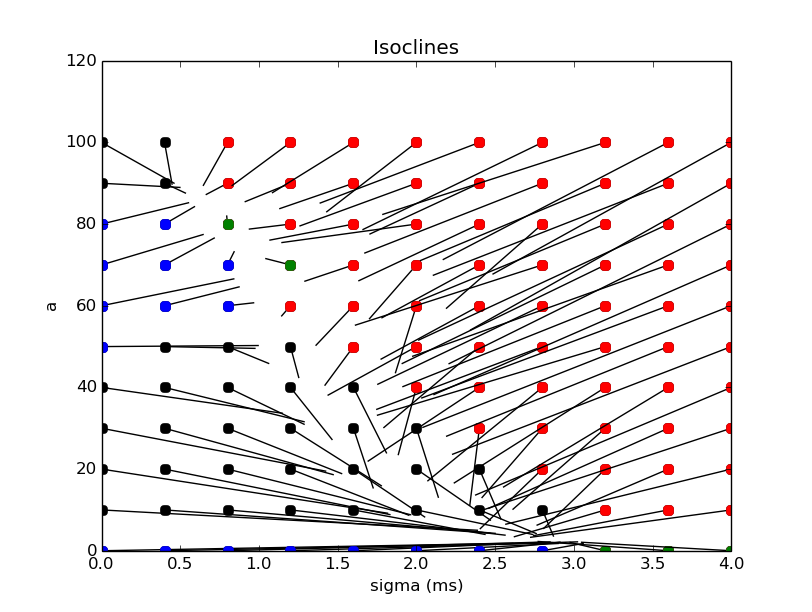
**Date - 19/08/2015**

1. Plot isoclines as a function of w. (Methods: Isoclines for figure 4(a,b,c,d))
2. Isoclines :  
   a. Sigma-isocline:   
    1. Not dependent on w.  
    2. Maintaining state of temporal spread irrespective of a.

b. a-isocline:  
 1. Dependent on w  
 2. Maintaining state of having a spike number which doesn’t change   
 with sigma.

1. Methods : ?
2. a- isocline is plotted by picking all those set of points for which the a\_out-a\_in == 0 or change is a is zero.
3. sigma-isocline is plotted by picking all those points for which sigma\_out-sigma\_in == 0.
4. Since it is being done on a limited number of grid points it is difficult to exactly arrive at the isoclines. So a coloring of the region is done the blue region is where the ‘a’ increases and red where the ‘sigma\_out’ decreases. The boundary of the two regions is where the isoclines lie.  
     
   The figure below is generated for a w of 100 (neuron\_multiplicity 50). Red region is all those a\_in,sigma\_in for which the sigma\_out decreases and Blue region is where a\_out increases. The boundary of each region marks the respective nullclines.
5. Plot for w = 80 (neuron\_multiplicity 50):
6. Plot for w = 90 (neuron\_multiplicity 50):
7. Plot for w = 100 (neuron\_multiplicity 50):
8. Plot for w = 110 (neuron\_multiplicity 50): 

**Date - 20/08/2015**

1. Improved the coloring algorithm to find out overlapping grid points by giving them a unique color.
2. Plot for w = 110 (neuron\_multiplicity 50)
3. Plot for w (neurons/layer) = 100 (neuron\_multiplicity 50)
4. Plot for w (neurons/layer) = 90 (neuron\_multiplicity 50)

**Date - 21/08/2015**

1. Update mid-term report with figures generated using brian and attach codes.