# It is not recommended for users to read this file. The main points are:

# A long mcmc (either ESS or MH) will often find a higher logP value than even a fine grid search

* For ESS and MH, it is important that the sampling not be too short. That is even more important for ESS than for MH. So for ESS, it makes most sense to not do grid searches unless one is willing to endure a high computational cost.

# As noted elsewhere, ESS can have some ugly outlier sampling and may require filtering out the low probability values to get nice distributions and reasonabl mu\_AP.

# When trying to get uncertainties on the posterior (by HPD sampling) the MH might sometimes be better for pathological cases. ESS seems to still do well at finding the MAP but not so much at finding the posterior covariance when there are multiple modes. While multiple modes presents a challenge for MH, MH can still find the posterior covariance relatively well compared to other methods.

# Example 3a (some base cases)

# 03a2 is MH sampling with length of 10000 and automatic other settings. This takes around 5 minutes. We see that the sampling is not yet very good, but that it’s already reasonably converged.

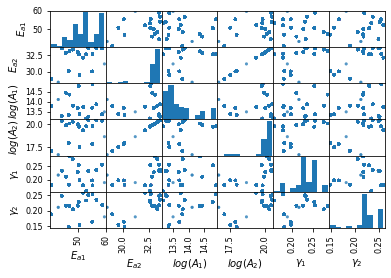
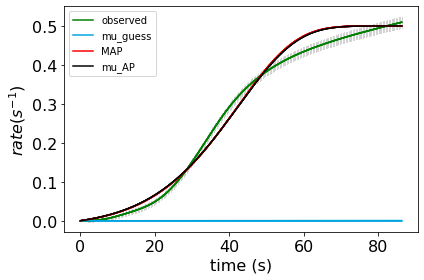
MAP\_logP:[376.10322722]

self.map\_index:5053

self.map\_parameter\_set:[53.83880958 33.60988329 13.24893493 20.45471553 0.23114804 0.25069333]

self.mu\_AP\_parameter\_set:[53.04865689 32.99455608 13.68928574 20.01424941 0.23400689 0.22458766]

self.stdap\_parameter\_set:[4.58423582 1.05906478 0.48373352 0.7577135 0.02635939 0.02275251]



# 03a3 is ESS sampling with length of 1000 and automatic other settings. This ends up with 24 walkers and 41 samplings. It’s around 1 hour of time and this can be one of our “base cases” to beat. We can see below that this worked reasonably well for the map, but retains some huge outliers. The sampling is probably a little better than it looks. By default, parameter\_estimation\_settings['mcmc\_threshold\_filter\_coefficient']= 'auto' gives a value of 2.0. Here, it might be worth running again with lower values (like 1.0 or 0.5).

MAP\_logP:[367.9318712]

self.map\_index:60

self.map\_parameter\_set:[3.25991227e+01 4.71041583e+01 1.97826485e+01 9.52367506e+00

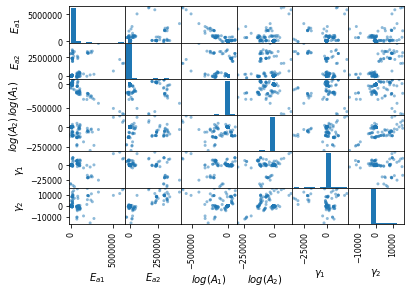
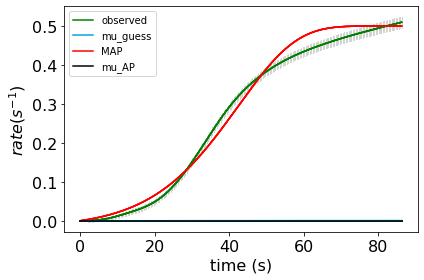
1.70761324e-02 1.31499016e-01]

self.mu\_AP\_parameter\_set:[ 9.86624272e+04 1.34521927e+05 -1.13267534e+04 -5.86123175e+03

-7.03222389e+01 2.38113652e+02]

self.stdap\_parameter\_set:[500762.99866394 584722.78860513 59071.913937 30253.89770259

3684.32887821 2176.98415695]



# 3a4 is ESS sampling with length of 10000 and automatic other settings. This ends up with 24 walkers and 416 samplings. It’s around 3 hours of time. So the scaling of ESS tends to get better with longer runs, unlike mcmc which tends to scale linearly with longer runs. We can see below that we get a higher map\_logP though in this system it is rather similar to the shorter run’s solution. The sampling is probably a little better than it looks. Using parameter\_estimation\_settings['mcmc\_threshold\_filter\_coefficient']= 'auto' gives a value of 2.0. Here, it might be worth running again with lower values (like 1.0 or 0.5). That is probably true for 03a3 as well.

MAP\_logP:[379.46672562]

self.map\_index:3635

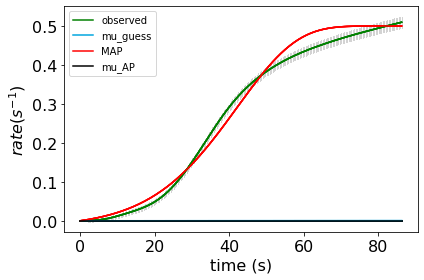
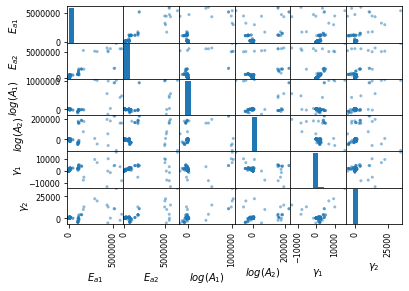
self.map\_parameter\_set:[44.41591492 34.09676656 12.1028402 20.7437047 0.12035418 0.49555743]

self.mu\_AP\_parameter\_set:[10356.71827596 11504.83895895 622.25041128 186.79959787

15.77854852 18.60634339]

self.stdap\_parameter\_set:[179268.5229734 195581.91256035 27239.67122764 7828.00019822

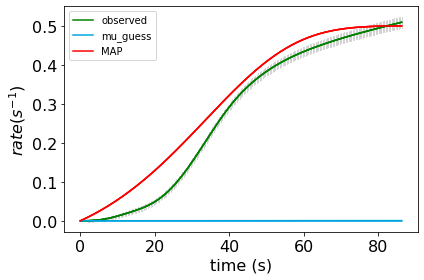
357.52555916 605.81618262]

# Example 03d (coarse gridsearch & gridsearch + w/ short mcmc)

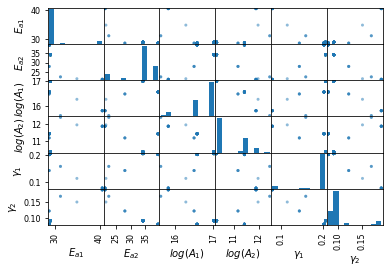
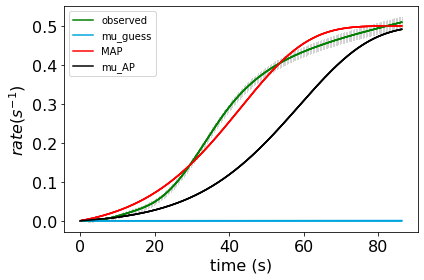
03d: Final map results from gridsearch: (20.0, 40, 11.0, 13.0, -1.4901161138336505e-09, 0.09999999850988389) final logP: -3169.1366742547816

This runs very fast (< 5 minutes) and does not do well. For more sampling one can see 3e which has a fine grid which does better.



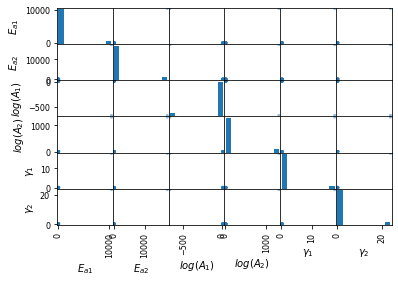
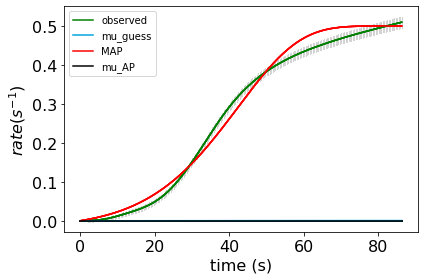
03d2 with burn\_in of 1 and MH mcmc length of 100 (this burn in length is too low, and this sampling length is also low, but we are mostly comparing the effects of gridsearch here). On the order of 3 seconds per mcmc sample. < 3 hours. A reasonable map is found.

Final map results from gridsearch: [28.89668578 34.14353553 17.01548942 10.32292743 0.20682889 0.09104755] final logP: [289.22086902]



03d3 ESS with burn\_in of ‘auto’ and mcmc\_length of 10. In this case, that is giving a burn in of 0 and 1 step since there are 12 walkers. The iterations are just 1 step (but 12 different at the same time) and are on the order of 20 seconds per iteration. Around 1 day.

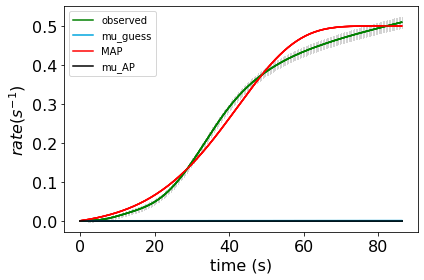
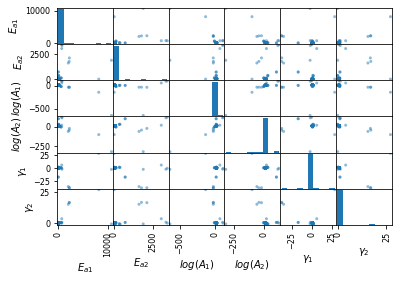
Final map results from gridsearch: [31.59174483 40.38037633 19.01113513 9.81237015 0.17348523 0.15534966] final logP: [365.72219032]



03d4 ESS with burn\_in of ‘auto’ and mcmc\_length of 100. In this case, that is giving 8 steps and 0 burn in per sampling, since there are 12 walkers (100/12 = ~8). Started at 10:00 AM: The iterations are not just 1 step and are on the order of 20 seconds per iteration. After around 2-3 days.

Final map results from gridsearch: [ 3.35098049e+01 5.20183899e+01 2.03400709e+01 1.24168798e+01

4.16775303e-01 -1.89881493e-02] final logP: [378.25748069]

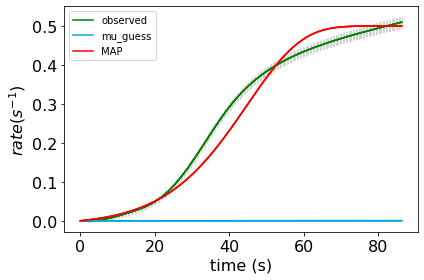
 

Overall, we see that for this problem the MetropolisHastings algorithm (regular mcmc) finds a good solution faster than the EnsembleSliceSampling does. This is probably mainly because the example is somewhat pathological. The test done above should not be taken to mean that the EnsembleSliceSampling is worse – just that it is *initially* slower to converge to a good solution. The ESS method keeps going ‘outwards’ from its starting points. Let’s do a longer sampling with ESS to make a more fair comparison:

# Example 03e: (finer gridsearch) vs. (coarser gridsearch w/ mcmc)

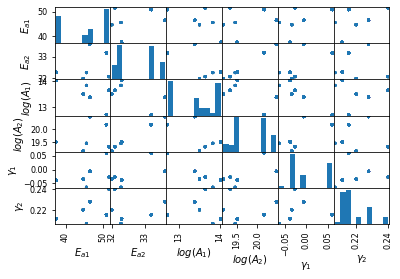
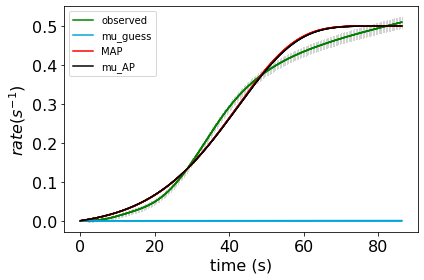
Example 3e, a fine grid:

Final map results from gridsearch: (44.0, 36.0, 13.0, 22.0, 0.1, 0.2) final logP: -44.74269267502692



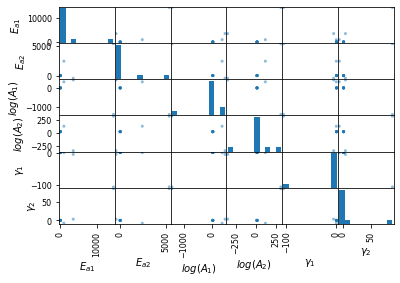
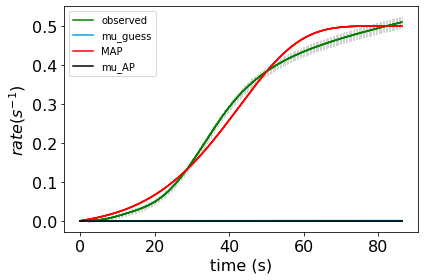
Example 3e2, a coarse grid MH mcmc with 1000 samples per gridpoint. ~ 1 hour total. Output below is similar to 3d2 (shorter mcmc) and also similar to 3a3 which is ESS with 1 hour of sampling. So for the same time of sampling we do see something similar.

Final map results from gridsearch: [50.84736639 33.1785014 12.71535263 20.15766407 0.0606503 0.21544758] final logP: [376.08249122]

/

03e3 a coarse grid ESS mcmc with burn\_in of ‘auto’ and mcmc\_length of 10. In this case, that is giving a burn in of 0 and 1 step since there are 12 walkers. The iterations are just 1 step (but 12 different at the same time) and are on the order of 20 seconds per iteration, 1-2 hours total. We see that even this short sampling, is not really any better than 1 hour of non-grid ESS (3a3) or of 1 hour of grid mcmc (3d2).

Final map results from gridsearch: [68.21053833 32.23895422 10.64485709 19.44069816 0.25394317 0.25728376] final logP: [366.92444066]



03e4 a coarse grid ESS mcmc with burn\_in of ‘auto’ and mcmc\_length of 100. In this case, that is giving a 8 steps with 12 walkers. 5-20 hours total.

Final map results from gridsearch: [57.17538519 33.34734491 14.65669952 20.28729524 0.19747315 0.16176287] final logP: [373.61673784]

