C:\Users\fvs\Documents\GitHub\CheKiPEUQ\Examples\Example15\runfile\_Example15a\_two\_site\_ratio\_d\_initial\_guess.py

Took the initial guess from a gridsearch (d) and then did mcmc using that as an initial guess. That gridsearch had a “reasonable” value, but the mcmc ended up pushing A1 into unrealistic values, despite starting at a realistic value.

~~Now in version 16, adding in the vertical offset which is going to be 0.005 and 0.001 as the standard deviation.~~

Changed back to using original file input with 0.005 uncertainty

Using this: C:\Users\fvs\Documents\GitHub\CheKiPEUQ\Examples\Example16\runfile\_Example15d\_two\_site\_ratio.py

With this:

PE\_object.doGridSearch('getLogP', gridSamplingAbsoluteIntervalSize=np.array(UserInput.model['InputParametersPriorValuesUncertainties']), gridSamplingNumOfIntervals=[2,2,2,2,2,2,2], verbose = True)

Which takes 10 minutes. Apparently below would only take 1 hour. So could make a finer grid.

PE\_object.doGridSearch('getLogP', gridSamplingAbsoluteIntervalSize=np.array(UserInput.model['InputParametersPriorValuesUncertainties']), gridSamplingNumOfIntervals=[3,3,3,3,3,2,2], verbose = True)

Will do this….

gridSamplingAbsoluteIntervalSize = [0.50/3, 10, 10, 1,1, 0.1, 0.1]

gridSamplingNumOfIntervals=[3,5,5,5,5,2,2], which is around 10 hours.

Had this result, which is concerning because of the 18 being at he edge of the allowed range.

0.33333333333333337, 21.5, 31.5, 12.0, 18.0, -0.1, 0.30000000000000004

Doing 20,000 mcmc samplings in C:\Users\fvs\Documents\GitHub\CheKiPEUQ\Examples\Example16\runfile\_Example15a\_two\_site\_ratio\_d\_initial\_guess.py

MAP\_logP:[153.83066979]

self.map\_index:16325

self.map\_parameter\_set:[ 0.25621994 24.18377348 37.56220456 13.54975894 23.9337553 -0.23260353

0.32934818]

self.mu\_AP\_parameter\_set:[ 0.28020449 21.98859657 34.34588062 12.04392328 21.309083 -0.20579051

0.37237914]

self.stdap\_parameter\_set:[0.03286409 1.02479803 1.72291198 0.72492619 1.30577351 0.03797193

0.05300712]

Now doing 200,000 samplings to see if the pre-exponential 2 of 21 goes even higher.

Basically, yes:

MAP\_logP:[590.58207571]

self.map\_index:163006

self.map\_parameter\_set:[ 0.73219435 31.28311851 45.33287731 17.41267305 29.34240026 -0.14982982

0.30270917]

self.mu\_AP\_parameter\_set:[ 0.77918148 25.65596376 41.13716493 13.53095376 26.17482362 -0.19787239

0.3180629 ]

self.stdap\_parameter\_set:[0.05653414 2.48512324 2.5346084 1.74610975 1.89415728 0.02530495

0.04507378]

Going to 9 parameter case with runfile\_Example16a\_two\_site\_NineParameters and processing\_functions\_tpd\_odeint\_two\_site\_NineParameters

Strangely, it now was taking more time to do the grid sampling with 9 parameters, 4-11 days.

Need to figure out why.

Also, this set of parmaeters from mcmc:

self.map\_parameter\_set:[ 1.10041633e+00 -2.86936286e-03 -7.59004532e-02 2.75992148e+01

3.14641581e+01 1.63898938e+01 1.43717723e+01 -1.80551162e-01

4.13798151e-01]

Produced a curve that went negative at the end. Need to see how it got a negative rate. Was it just due the background subtraction, or something else?

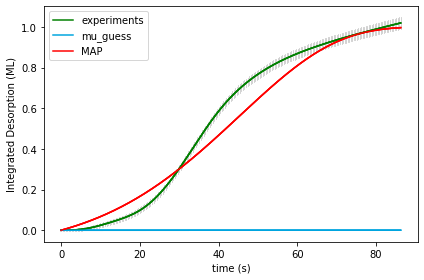
* Ok, eventually I figured out that what happened is that the ratio went to negative such that the second site has a negative concentration. This means it’s more important to have the absolute cutoffs.
* After adding a lower bound of 0, then the ratio even tried going over 1. So needed to put an upper bound of 1 also.

Now, with these bounds, using C:\Users\fvs\Documents\GitHub\CheKiPEUQ\Examples\Example16\runfile\_Example16a\_two\_site\_NineParameters\_grid\_reduced.py and PE\_object.doGridSearch('getLogP', gridSamplingAbsoluteIntervalSize=[0.03, 0.0025, 0.50/3, 10, 10, 1,1, 0.1, 0.1], gridSamplingNumOfIntervals=[0,0,2,4,4,4,4,2,2], verbose = True)

This has around 2 hours of waiting time, (apparently 800,000 points) and is reasonable for a screening.

This gave grid map of (1.0, 0.0, 0.5, 30.0, 20.0, 17.0, 11.0, 0.30000000000000004, 0.1)

It actually does not look like a very good match, but will start with it for MCMC and see what happens (below is what it looks like). It’s good that it looks like a bad match, that means it’s a local max on a prior.



Taking that as a starting point and using C:\Users\fvs\Documents\GitHub\CheKiPEUQ\Examples\Example16\runfile\_Example16a\_two\_site\_NineParameters.py

Sadly, this quickly resulted in terrible numbers. Turning off scaling uncertainties.

That changed the output, but the output still went into unrealistic territory.

Making the prior uncertainties like this seemed to work:

UserInput.model['InputParametersPriorValuesUncertainties'] = [ 0.10, 0.005, 0.50/3, 20, 20, 2, 2, 0.1, 0.1]