SIMPLE-ABC:
Statistical Inference for Multiple PLanet systems Employing Approximate Bayesian Computation

Robert C. Morehead

The Pennsylvania State University

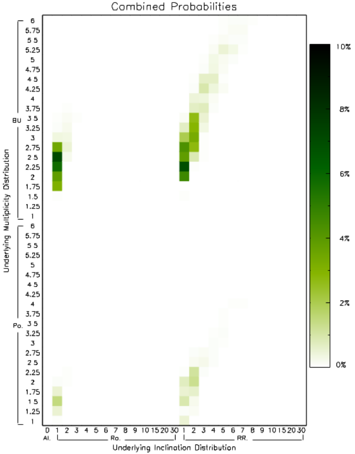
04/30/2014

We would like to make inferences about the underlying distribution(s) of *Kepler* Planets.

-What is the number of planets per star?

-What is the distribution of mutual inclinations?

-What is the distribution of planet eccentricities?



Fang & Margot (2012)

We want to infer the posterior probability of some model with parameters θ.

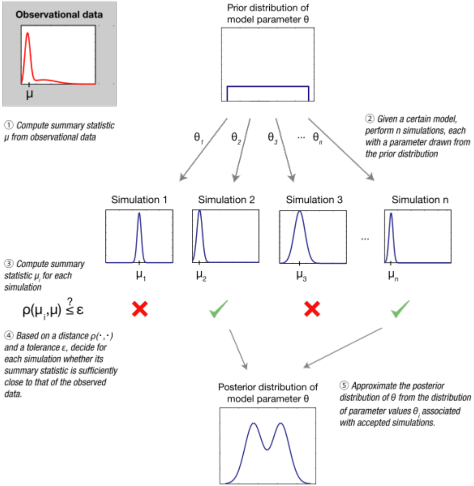
Posterior

Prior

Likelihood

$$\Pr(\theta \mid \text{Data}) = \frac{\Pr(\theta)\,\Pr(\text{Data} \mid \theta)}{\Pr(\text{Data})}$$

Approximate Bayesian Computation (ABC) is a likelihood-free method to infer posterior distributions.



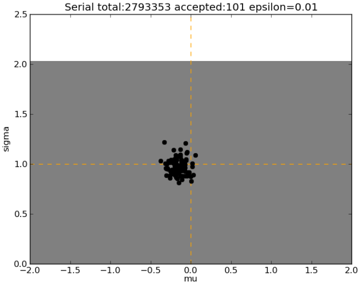
Sunnåker et al. (2013)

I took an object-oriented approach using Python.

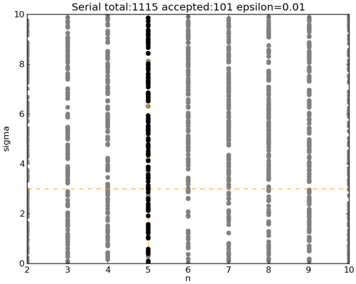
```
class Model(object):

    def __call__(self, theta):
        return self.generate_data_and_reduce(theta)

    def set_data(self, data):
            self.data = data
            self.data_sum_stats = self.summary_stats(self.data)

    def generate_data_and_reduce(self, theta):
        """
        A combined method for generating data, calculating summary statistics
        and evaluating the distance function all at once.
        """
        synth = self.generate_data(theta)
        sum_stats = self.summary_stats(synth)
        d = self.distance_function(sum_stats, self.data_sum_stats)
        return d

    def draw_theta(self):
        raise NotImplementedError('You must override the draw_theta '
                                  'method in your own subclass.')

    def generate_data(self, theta):
        raise NotImplementedError('You must override the generate_data '
                                  'method in your own subclass.')

    def summary_stats(self, data):
        raise NotImplementedError('You must override the summary_stats '
                                  'method in your own subclass.')

    def distance_function(self, summary_stats, summary_stats_synth):
        raise NotImplementedError('You must override the distance_function '
                                  'method in your own subclass.')
```

For testing purposes, I implemented two models.

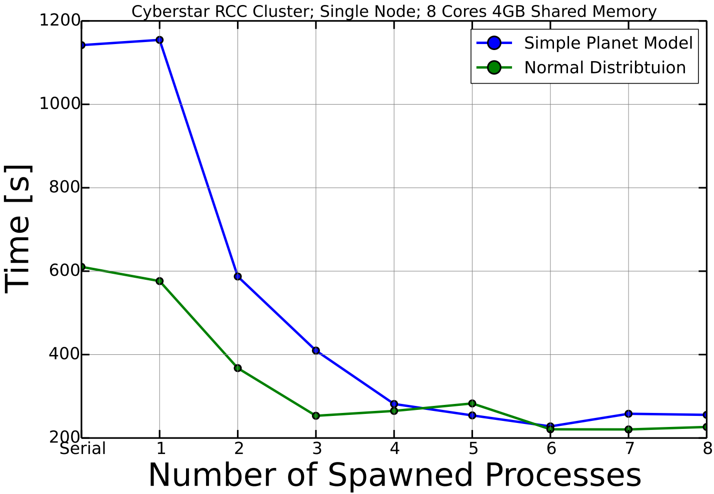
Normal Distribution



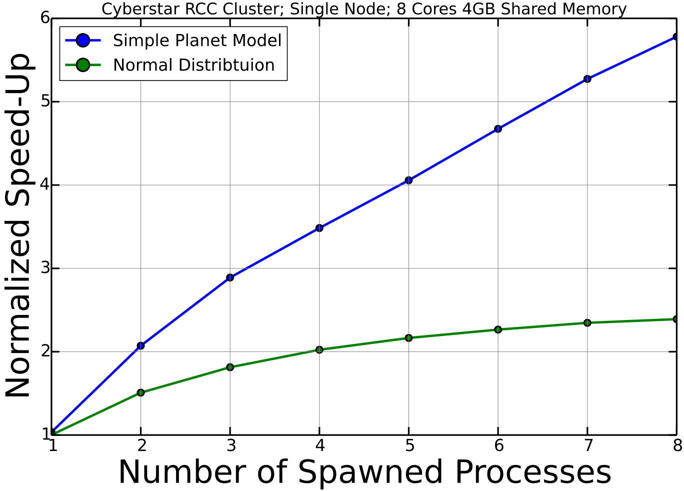
Simple Planet Model



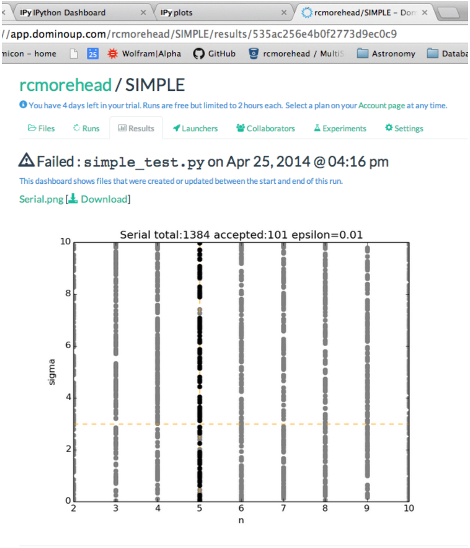
Parallelization via multiprocessing.pool.map works pretty well...



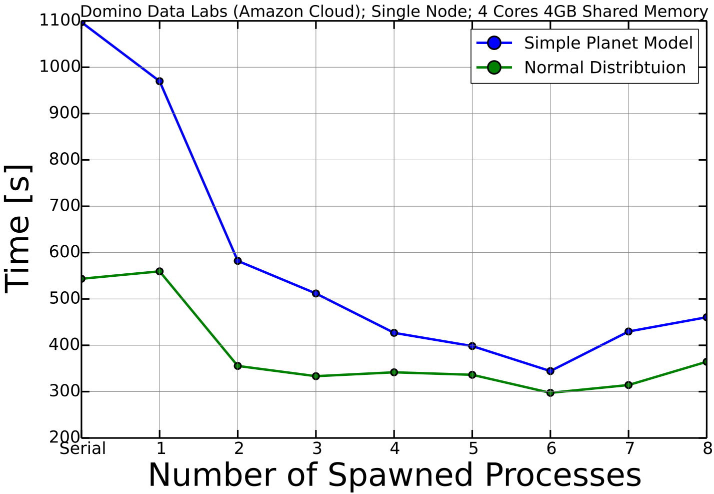
and even better for complex models!



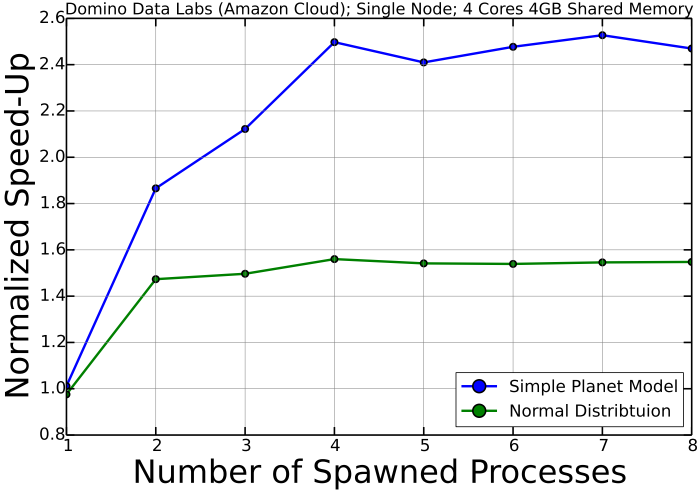
I tried cloud computing with Domino Data Labs.



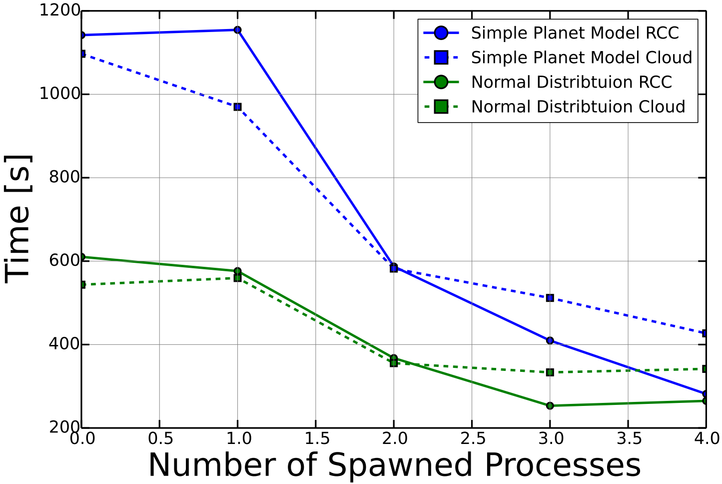
If I had presented on Monday, that would have been it...



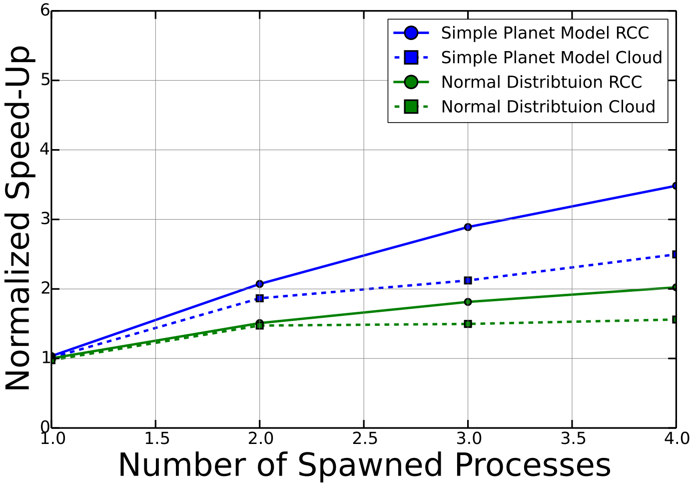
The gains in performance are *almost* as good.



But in the cloud system does not scale quite as well.



But in the cloud system does not scale quite as well.



The End

Nothing to see here!

For testing purposes, I implemented two models.

Normal Distribution

$\theta = (\mu, \sigma)$

$(\mu = 0, \sigma = 1)$

$\widehat{S} = (\text{25th, 75th percentiles})$

$D = \sqrt{\widehat{S}_0^2 + \widehat{S}_1^2}$

$\epsilon = 0.01$

Minimum particles $= 100$

Simple Planet Model

$\theta = (n, \sigma)$

$(n = 5, \sigma = 3)$

$\widehat{S} = (\text{N transits per star}, \ b)$

$D = \sqrt{D_{KS}(\widehat{S}_0)^2 + D_{KS}(\widehat{S}_1)^2}$

$\epsilon = 0.01$

Minimum particles $= 100$

Fang, J., & Margot, J.-L. 2012, , 761, 92

Sunnåker, M., Busetto, A. G., Numminen, E., Corander, J., Foll, M., & Dessimoz, C. 2013, PLoS Comput Biol, 9, e1002803