
dessn Documentation

Release 0.0.1

dessn

March 03, 2016

1	dessn package	1
1.1	Subpackages	1
1.1.1	dessn.entry package	1
	Submodules	1
	dessn.entry.sim module	1
1.1.2	dessn.model package	1
	Subpackages	1
	Submodules	1
	dessn.model.abstracts module	1
	dessn.model.model module	1
	dessn.model.parameter module	1
1.1.3	dessn.simple package	1
	Submodules	2
	dessn.simple.example module	2
	dessn.simple.exampleIntegral module	3
	dessn.simple.exampleLatent module	5
	dessn.simple.exampleLatentClass module	7
1.1.4	dessn.simulation package	7
	Submodules	7
	dessn.simulation.observationFactory module	7
	dessn.simulation.simulation module	8
	Python Module Index	9
	Index	11

DESSN PACKAGE

Welcome to the **DESSN** code base.

1.1 Subpackages

1.1.1 `dessn.entry` package

Submodules

`dessn.entry.sim` module

1.1.2 `dessn.model` package

Subpackages

`dessn.model.nodes` package

Submodules

`dessn.model.nodes.cosmology` module

`dessn.model.nodes.supernova` module

`dessn.model.nodes.typeProb` module

Submodules

`dessn.model.abstracts` module

`dessn.model.model` module

`dessn.model.parameter` module

1.1.3 `dessn.simple` package

This module is designed to give a step by step overview of a very simplified example Bayesian model.

The basic example model is laid out in the parent class `Example`, and there are three implementations. The first implementation, `ExampleIntegral`, shows how the problem might be approached in a simple model, where numerical integration is simply done as part of the likelihood calculation.

However, if there are multiple latent parameters, we get polynomial growth of the number of numerical integrations we have to do, and so this does not scale well at all.

This leads us to the implementation in `ExampleLatent`, where we use the MCMC algorithm to essentially do Monte Carlo integration via marginalisation. This means we do not need to perform the numerical integration in the likelihood calculation, however the cost of doing so is increase dimensionality of our MCMC.

Finally, the `ExampleLatentClass` implementation shows how the `ExampleLatent` class might be written to make use of Nodes. This is done in preparation for more complicated models, which will have more than one layer and needs to be configurable.

Submodules

dessn.simple.example module

`class dessn.simple.example.Example (n=300, theta_1=100.0, theta_2=30.0)`

Bases: `object`

Setting up the math for some examples.

Let us assume that we are observing supernova that are drawn from an underlying supernova distribution parameterised by θ , where the supernova itself simply a luminosity L . We measure the luminosity of multiple supernovas, giving us an array of measurements D . If we wish to recover the underlying distribution of supernovas from our measurements, we wish to find $P(\theta|D)$, which is given by

$$P(\theta|D) \propto P(D|\theta)P(\theta)$$

Note that in the above equation, we realise that $P(D|L) = \prod_{i=1}^N P(D_i|L_i)$ as our measurements are independent. The likelihood $P(D|\theta)$ is given by

$$P(D|\theta) = \prod_{i=1}^N \int_{-\infty}^{\infty} P(D_i|L_i)P(L_i|\theta)dL_i$$

We now have two distributions to characterise. Let us assume both are gaussian, that is our observed luminosity x_i has gaussian error σ_i from the actual supernova luminosity, and the supernova luminosity is drawn from an underlying gaussian distribution parameterised by θ .

$$P(D_i|L_i) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left(-\frac{(x_i - L_i)^2}{\sigma_i^2}\right)$$

$$P(L_i|\theta) = \frac{1}{\sqrt{2\pi}\theta_2} \exp\left(-\frac{(L_i - \theta_1)^2}{\theta_2^2}\right)$$

This gives us a likelihood of

$$P(D|\theta) = \prod_{i=1}^N \frac{1}{2\pi\theta_2\sigma_i} \int_{-\infty}^{\infty} \exp\left(-\frac{(x_i - L_i)^2}{\sigma_i^2} - \frac{(L_i - \theta_1)^2}{\theta_2^2}\right) dL_i$$

Working in log space for as much as possible will assist in numerical precision, so we can rewrite this as

$$\log(P(D|\theta)) = \sum_{i=1}^N \left[\log\left(\int_{-\infty}^{\infty} \exp\left(-\frac{(x_i - L_i)^2}{\sigma_i^2} - \frac{(L_i - \theta_1)^2}{\theta_2^2}\right) dL_i\right) - \log(2\pi\theta_2\sigma_i) \right]$$

Parameters **n** : int, optional
 The number of supernova to ‘observe’

theta_1 : float, optional
 The mean of the underlying supernova luminosity distribution

theta_2 : float, optional
 The standard deviation of the underlying supernova luminosity distribution

do_emcee (*nwalkers=None, nburn=None, nsteps=None*)
 Abstract method to configure the emcee parameters

get_likelihood (*theta, data, error*)
 Abstract method to return the log likelihood

get_posterior (*theta, data, error*)
 Gives the log posterior probability given the supplied input parameters.

Parameters **theta** : array of model parameters

data : array of length *n*
 An array of observed luminosities

error : array of length *n*
 An array of observed luminosity errors

Returns float
 the log posterior probability

get_prior (*theta*)
 Get the log prior probability given the input.
 The prior distribution is currently implemented as flat prior.

Parameters **theta** : array of model parameters

Returns float
 the log prior probability

plot_observations ()
 Plot the observations and observation distribution.

dessn.simple.exampleIntegral module

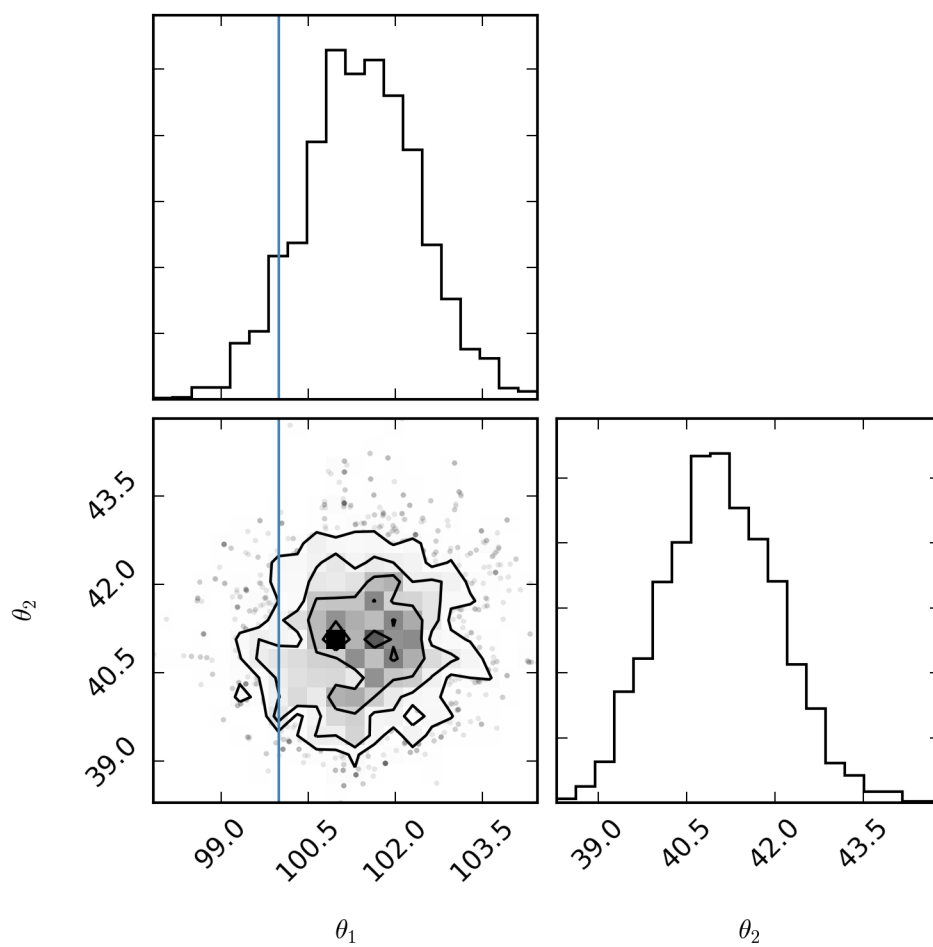
class `dessn.simple.exampleIntegral.ExampleIntegral` (*n=900, theta_1=100.0, theta_2=30.0*)

Bases: `dessn.simple.example.Example`

An example implementation using integration over a latent parameter.

Building off the math from [Example](#) Creating this class will set up observations from an underlying distribution. Invoke `emcee` by calling the object. In this example, we perform the marginalisation inside the likelihood calculation, which gives us dimensionality only of two (the length of the θ array). However, this is at the expense of performing the marginalisation over dL_i , as this requires computing n integrals for each step in the MCMC.

Parameters **n** : int, optional
 The number of supernova to ‘observe’



theta_1 : float, optional

The mean of the underlying supernova luminosity distribution

theta_2 : float, optional

The standard deviation of the underlying supernova luminosity distribution

do_emcee (*nwalkers=20, nburn=500, nsteps=2000*)

Run the *emcee* chain and produce a corner plot.

Saves a png image of the corner plot to plots/exampleIntegration.png.

Parameters **nwalkers** : int, optional

The number of walkers to use. Minimum of four.

nburn : int, optional

The burn in period of the chains.

nsteps : int, optional

The number of steps to run

get_likelihood (*theta, data, error*)

Gets the log likelihood given the supplied input parameters.

Parameters **theta** : array of size 2

An array representing $[\theta_1, \theta_2]$

data : array of length n

An array of observed luminosities

error : array of length n

An array of observed luminosity errors

Returns float

the log likelihood probability

dessn.simple.exampleLatent module

class `dessn.simple.exampleLatent.ExampleLatent` (*n=300, theta_1=100.0, theta_2=30.0*)

Bases: `dessn.simple.example.Example`

An example implementation using marginalisation over latent parameters.

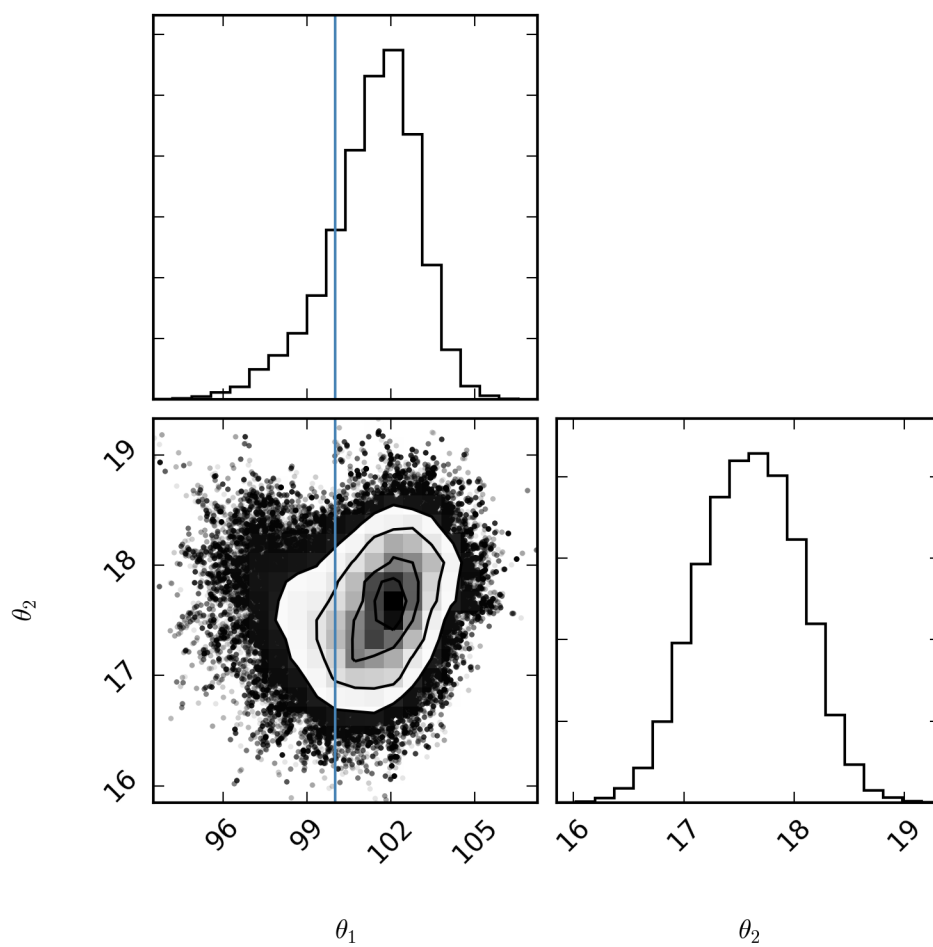
Building off the math from [Example](#), instead of performing the integration numerically in the computation of the likelihood, we can instead use Monte Carlo integration by simply setting the latent parameters \vec{L} as free parameters, giving us

$$\log \left(P(D|\theta, \vec{L}) \right) = - \sum_{i=1}^N \left[\frac{(x_i - L_i)^2}{\sigma_i^2} + \frac{(L_i - \theta_1)^2}{\theta_2^2} + \log(2\pi\theta_2\sigma_i) \right]$$

Creating this class will set up observations from an underlying distribution. Invoke `emcee` by calling the object. In this example, we marginalise over L_i after running our MCMC, and so we no longer have to compute integrals in our chain, but instead have dimensionality of $2 + n$, where n are the number of observations.

Parameters **n** : int, optional

The number of supernova to ‘observe’



theta_1 : float, optional

The mean of the underlying supernova luminosity distribution

theta_2 : float, optional

The standard deviation of the underlying supernova luminosity distribution

do_emcee (*nwalkers=200, nburn=1000, nsteps=4000*)

Run the *emcee* chain and produce a corner plot.

Saves a png image of the corner plot to plots/exampleLatent.png.

Parameters **nwalkers** : int, optional

The number of walkers to use.

nburn : int, optional

The burn in period of the chains.

nsteps : int, optional

The number of steps to run

get_likelihood (*theta, data, error*)

Gets the log likelihood given the supplied input parameters.

Parameters **theta** : array of length $2 + n$

An array representing $[\theta_1, \theta_2, \vec{L}]$

data : array of length n

An array of observed luminosities

error : array of length n

An array of observed luminosity errors

Returns float

the log likelihood probability

dessn.simple.exampleLatentClass module

1.1.4 dessn.simulation package

Submodules

dessn.simulation.observationFactory module

class `dessn.simulation.observationFactory.ObservationFactory` (***kwargs*)

Bases: `object`

check_kwargs ()

get_observations (*num*)

Still needs massive refactoring

dessn.simulation.simulation module

```
class dessn.simulation.simulation.Simulation  
    Bases: object  
    get_simulation (num_trans=30)
```

d

- `dessn`, 1
- `dessn.entry`, 1
- `dessn.entry.sim`, 1
- `dessn.model`, 1
- `dessn.model.nodes`, 1
- `dessn.simple`, 1
- `dessn.simple.example`, 2
- `dessn.simple.exampleIntegral`, 3
- `dessn.simple.exampleLatent`, 5
- `dessn.simulation`, 7
- `dessn.simulation.observationFactory`, 7
- `dessn.simulation.simulation`, 8

C

check_kwargs() (dessn.simulation.observationFactory.ObservationFactory
method), 7

D

dessn (module), 1
 dessn.entry (module), 1
 dessn.entry.sim (module), 1
 dessn.model (module), 1
 dessn.model.nodes (module), 1
 dessn.simple (module), 1
 dessn.simple.example (module), 2
 dessn.simple.exampleIntegral (module), 3
 dessn.simple.exampleLatent (module), 5
 dessn.simulation (module), 7
 dessn.simulation.observationFactory (module), 7
 dessn.simulation.simulation (module), 8
 do_emcee() (dessn.simple.example.Example method), 3
 do_emcee() (dessn.simple.exampleIntegral.ExampleIntegral
method), 5
 do_emcee() (dessn.simple.exampleLatent.ExampleLatent
method), 7

E

Example (class in dessn.simple.example), 2
 ExampleIntegral (class in dessn.simple.exampleIntegral),
3
 ExampleLatent (class in dessn.simple.exampleLatent), 5

G

get_likelihood() (dessn.simple.example.Example
method), 3
 get_likelihood() (dessn.simple.exampleIntegral.ExampleIntegral
method), 5
 get_likelihood() (dessn.simple.exampleLatent.ExampleLatent
method), 7
 get_observations() (dessn.simulation.observationFactory.ObservationFactory
method), 7
 get_posterior() (dessn.simple.example.Example method),
3
 get_prior() (dessn.simple.example.Example method), 3

get_simulation() (dessn.simulation.simulation.Simulation
method), 8

O

ObservationFactory (class in
dessn.simulation.observationFactory), 7

P

plot_observations() (dessn.simple.example.Example
method), 3

S

Simulation (class in dessn.simulation.simulation), 8