learning - exercises

1. Our prior beliefs about coins

a)

Recall our final coin weight model, “fair-vs-uniform”, in which the coin weight was either 0.5 with high probability or drawn from a uniform distribution otherwise. This implies that a two-faced coin (always heads) is equally likely as a 70% heads coin. Intuitively you might be inclined to think that a two-faced coin is easier to make, and thus more likely. Adjust the model to express this prior.

```
x
```

```
var weightPosterior = function(observedData){
```

```
  return Infer({method: 'MCMC', burn:1000, samples: 10000}, function() {
```

```
    //
```

```
    var realWeight = //..your code here
```

var realWeight = flip(.999) ? 0.5 : 1;

```
    var coin = Bernoulli({p: realWeight})
```

```
    var obsFn = function(datum){observe(coin, datum=='h')}
```

```
    mapData({data: observedData}, obsFn)
```

```
    return realWeight
```

```
  })
```

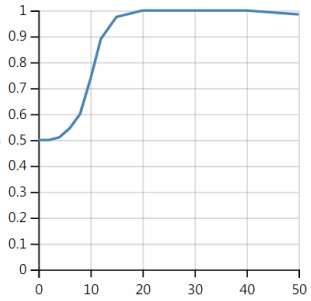
```
}
```

```

```

```
var fullDataSet = repeat(50, function(){return 'h'});
```

```
var observedDataSizes = [0,1,2,4,6,8,10,12,15,20,25,30,40,50];
```

```
var estimates = map(function(N) {
  return expectation(weightPosterior(fullDataSet.slice(0,N)))
}, observedDataSizes);

viz.line(observedDataSizes, estimates);
```



run ▼

b)

How does your solution behave differently than the fair-vs-uniform model from the chapter? Find a data set such that the learning curves are qualitatively different.

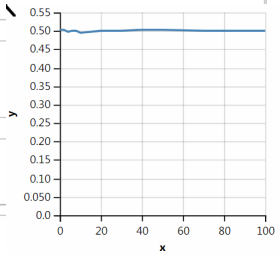
2. The strength of beliefs

a)

In the chapter, we graphed *learning trajectories* for a number of models. Below is one of these models (the one with the `Beta(10,10)` prior). In the chapter, we observed how the model's best guess about the weight of the coin changed across a sequence of sucessive heads. See what happens if instead we see heads and tails in alternation:

```
var pseudoCounts = {a: 10, b: 10};

var weightPosterior = function(observedData){
  return Infer({method: 'MCMC', burn:1000, samples: 1000}, function() {
    var coinWeight = beta(pseudoCounts)
    var coin = Bernoulli({p: coinWeight})
    var obsFn = function(datum){observe(coin, datum=='h')}
    mapData({data: observedData}, obsFn)
    return coinWeight
  })
}
```

```
//creating 50 pairs of 'h' and 't' alternating
var fullDataSet = repeat(50,function(){['h', 't']}).flat()


var observedDataSizes = [0,2,4,6,8,10,20,30,40,50,70,100]
var estimates = map(function(N) {
  return expectation(weightPosterior(fullDataSet.slice(0,N)))
}, observedDataSizes)
viz.line(observedDataSizes, estimates);
```



run ▼

It looks like we haven't learned anything! Indeed, since our best estimate for the coin's weight was 0.5 *prior* to observing anything, our best estimate is hardly going to change when we get data consistent with that prior.

However, we've been looking at the average (or expected) estimate. Edit the code below to see whether our posterior *distribution* is at all changed by observing this data set. (You only need to compare the prior and the posterior after all 100 observations):

```
var pseudoCounts = {a: 10, b: 10};


var weightPosterior = function(observedData){
  return Infer({method: 'MCMC', burn:1000, samples: 1000}, function() {
    var coinWeight = beta(pseudoCounts))
    var coin = Bernoulli({p: coinWeight})
    var obsFn = function(datum){observe(coin, datum=='h')}
    mapData({data: observedData}, obsFn)
```

```
    return coinWeight

  })

}


//creating 50 pairs of 'h' and 't' alternating

var fullDataSet = repeat(50,function(){['h', 't']}).flat()


var prior = //your code here
var post = //your code here


viz(prior) //should graph the prior distribution on weights

viz(post) //should graph the posterior distribution on weights
```

```
var prior = Beta(pseudoCounts)
var post = weightPosterior(fullDataSet)
```

run ▼ The uncertainty(the variance of the distribution) is reduced.

Based on your results, is anything learned? (Note that the bounds on the x-axis are likely to be different in the two graphs, which could obscure this. The `viz` package doesn't easily to allow you to adjust the bounds on the axes.)

b)

Ideally, we'd like to see how our belief distribution shifts as more data comes in. A particularly good measure would be entropy. Unfortunately, calculating entropy for a Beta distribution is somewhat involved (https://en.wikipedia.org/wiki/Beta_distribution#Quantities_of_information_(entropy)).

An alternative we can use is variance: the expected squared difference between a sample from the distribution and the distribution mean. This doesn't take into account the shape of the distribution, and so won't give us quite what we want if the distribution is non-symmetric; but it is a reasonable first try.

Edit the code below to see how variance changes as more data is observed.

```
var pseudoCounts = {a: 10, b: 10};
```

```
var weightPosterior = function(observedData){
  return Infer({method: 'MCMC', burn:1000, samples: 1000}, function() {
    var coinWeight = beta(pseudoCounts))
    var coin = Bernoulli({p: coinWeight})
    var obsFn = function(datum){observe(coin, datum=='h')}
    mapData({data: observedData}, obsFn)
    return coinWeight
  })
}

//creating 50 pairs of 'h' and 't' alternating
var fullDataSet = repeat(50,function(){['h', 't']}).flat()

var observedDataSizes = [0,2,4,8,16,32,64,128,256,512];
var posts = map(function(N) {
  return weightPosterior(fullDataSet.slice(0,N))
}, observedDataSizes);
// returns an array of posteriors of length observedDataSizes.length
```

```
var variances = map(function(p){

  // your code here

}, posts)

viz.line(observedDataSizes, variances);
```

```
var variances = map(function(p){
  var mean = expectation(p)
  return expectation(p, function(S){return Math.pow(S-mean, 2)})
}, posts)
```

run ▼

HINT: notice how the variable `posts` differs from `estimates` in the code above.

3. Causal Power

Consider our model of causal power from the chapter:

```
var observedData = [{C:true, E:false}]

var causalPowerPost = Infer({method: 'MCMC', samples: 10000, lag:2}, function() {
  // Causal power of C to cause E
  var cp = uniform(0, 1)

  // Background probability of E
  var b = uniform(0, 1)

  mapData({data: observedData}, function(datum) {
    // The noisy causal relation to get E given C
    var E = (datum.C && flip(cp)) || flip(b)
    condition(E == datum.E)
```

```
  })

  return {causal_power: cp, background: b}
});

viz.marginals(causalPowerPost);
```

run ▼

a)

Find a set of observations that result in inferring a fairly high causal power for C and a low background probability of E. Explain why this works. [{C:true, E:true},{C:false, E:false}]

b)

Find a set of observations that result in inferring a fairly low causal power for C and a high background probability of E. Explain why this works. [{C:false, E:true},{C:false, E:true}]

c)

Find a set of observations that result in inferring a fairly high causal power for C and a high background probability of E. Explain why this works. [{C:true, E:true},{C:true, E:true}]

d)

Suppose every time C is present, so is the effect E. Suppose C is present at least 5 times. Is there a way to nonetheless fail to infer a high causal power for C?

[{C:true, E:true}, {C:true, E:true}, {C:true, E:true}, {C:true, E:true}, {C:true, E:true}, {C:false, E:true}, {C:false, E:true}, {C:false, E:true}]