VE414 Lecture 8

Jing Liu

UM-SJTU Joint Institute

September 26, 2019

- We have considered both deterministic and distributional approximations for

$$\int_{\infty}^{\infty} f_{Y|X}\left(y \mid x\right)\, dy$$

now we turn to the 3rd approach, which is what people use often nowadays.

Q: Given a needle, a ruler, a pen and a piece of paper, how would you estimate

$$\pi = 3.14159265358979323846264338327950288419716939937510?$$

Q: Given a needle of length l dropped on a piece of paper with parallel lines t units apart, what is the probability that the needle will land across a line?

- Let y be the distance from the centre of the needle to the closest line, and let θ be the acute angle between the needle and the lines, then

$$f_Y(y) = \begin{cases} \dfrac{2}{t}, & \text{for } 0 \le y \le \dfrac{t}{2}\ , \\ 0, & \text{otherwise.} \end{cases} \quad \text{and} \quad f_\theta(\theta) = \begin{cases} \dfrac{2}{\pi}, & \text{for } 0 \le \theta \le \dfrac{\pi}{2}\ , \\ 0, & \text{otherwise.} \end{cases}$$

- The two random variables, Y and θ, are independent, so the joint pdf is

$$f_{Y,\theta}(y,\theta) = \begin{cases} \dfrac{4}{t\pi}, & \text{for } 0 \le y \le \dfrac{t}{2},\ 0 \le \theta \le \dfrac{\pi}{2} \\ 0, & \text{otherwise.} \end{cases},$$

- The needle crosses a line if

$$y \le \frac{l}{2}\sin\theta$$

- Assume $l < t$, then the probability is given by

$$p = \int_0^{\pi/2} \int_0^{l/2\sin\theta} \frac{4}{t\pi}\, dy\, d\theta = \frac{2l}{t\pi}$$

- Back to our question of estimating π, how can we make use of this?

$$X_k \implies \pi = \lim_{k\to\infty} \frac{2l}{t}\frac{k}{x_k}$$

- Using needle dropping to estimate π illustrates the essence of

  Monte Carlo simulation

- By dropping the needle repeatedly, we have obtained the information on

$$y \leq \frac{l}{2} \sin\theta$$

  in terms of x_k number of success out of k number of trials, which leads to

$$\frac{x_k}{k} \approx p = \int_0^{\pi/2} \int_0^{l/2\sin\theta} \frac{4}{t\pi}\, dy\, d\theta = \frac{2l}{t\pi}$$

- In general, the essence of the above idea is applied to the following integral

$$A = \int_{\mathcal{D}} f_{\mathbf{X}|\mathbf{Y}}\left(\mathbf{x} \mid \mathbf{y}\right) f_{\mathbf{Y}}\left(\mathbf{y}\right)\, d\mathbf{y}$$

  where $\mathcal{D}$ is region in high-dimensional space.

- If we can draw independent and identically distributed samples

$$\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(m)}$$

uniformly from the region $\mathcal{D}$, then the normalisation constant given by

$$\hat{A}_m = \frac{1}{m}\left[f_{\mathbf{X}|\mathbf{Y}}\left(\mathbf{x} \mid \mathbf{y}^{(1)}\right) f_{\mathbf{Y}}\left(\mathbf{y}^{(1)}\right) + \cdots + f_{\mathbf{X}|\mathbf{Y}}\left(\mathbf{x} \mid \mathbf{y}^{(m)}\right) f_{\mathbf{Y}}\left(\mathbf{y}^{(m)}\right)\right]$$

- The law of large numbers states the average of many independent random variables with common mean and finite variances tends to stabilise at their common mean; that is,

$$\lim_{m\to\infty} \hat{A}_m = A, \qquad \text{with probability 1.}$$

- Its convergence rate can be assessed by the central limit theorem,

$$\sqrt{m}\left(\hat{A}_m - A\right) \to \text{Normal}\left(0, \sigma^2\right), \qquad \text{in distribution}$$

where $\sigma^2 = \text{Var}\left(f_{\mathbf{X}|\mathbf{Y}}\left(\mathbf{x} \mid \mathbf{y}\right) f_{\mathbf{Y}}\left(\mathbf{y}\right)\right)$ in terms of $\mathbf{Y}$.

- Like normal approximation, the "error term" of Monte Carlo approximation

$$\hat{A}_m$$

depends on the number of points in your sample, i.e.

$$m$$

regardless of the dimensionality of $\mathcal{D}$, that is, a high-dimension integral can be approximated just as efficient as a one-dimensional integral.

- This makes Monte Carlo very appealing, because Gauss quadrature is highly efficient in 1-dimension, but its counterpart in high-dimensional space is not, i.e. the error depends on the dimension of $\mathcal{D}$.
- Another aspect of Monte Carlo is its simplicity or convenience when comes to various inference that rooted on posterior distribution, e.g. consider

$$\mathbb{E}\left[\mathbf{Y} \mid \mathbf{X}\right] = \int_{\mathcal{D}} \mathbf{Y} f_{\mathbf{X}|\mathbf{Y}}\left(\mathbf{x} \mid \mathbf{y}\right) f_{\mathbf{Y}}\left(\mathbf{y}\right) \, d\mathbf{y}$$

- In fact, the focus of using Monte Carlo is not about the normalising constant

$$A = \int_{\mathcal{D}} f_{\mathbf{X}|\mathbf{Y}}\left(\mathbf{x} \mid \mathbf{y}\right) f_{\mathbf{Y}}\left(\mathbf{y}\right)\, d\mathbf{y}$$

- It provides inference directly, to illustrate that, consider the following again

$$\begin{aligned} P &\sim \text{Uniform}\,(0,1) \\ X_k \mid P &\sim \text{Binomial}\,(k,p) \\ \implies P \mid X_3 = 2 &\sim \text{Beta}\,(3,2) \end{aligned}$$

- The easiest sampling method is based discretising on a grid approximation,

```
> # Given
> k = 3; x = 2; a = 0; b = 1;
> # No. nodes
> n = 10;
> # Define grid
> p_grid = seq(from=a, to=b, length.out=n)
```

```
> # define prior
> prior = rep(1, n)

> # compute likelihood at each value in grid
> likelihood = dbinom(x , size=k, prob=p_grid)

> # compute product of likelihood and prior
> unstd.posterior = likelihood * prior
```

- Notice the following step is different from deterministic approximation

```
> # discretising the distribution, so it sums to 1
> posterior = unstd.posterior / sum(unstd.posterior)
```

- Once we have the discretisation, we sample m values from it

```
> # sample size
> m = 10
```

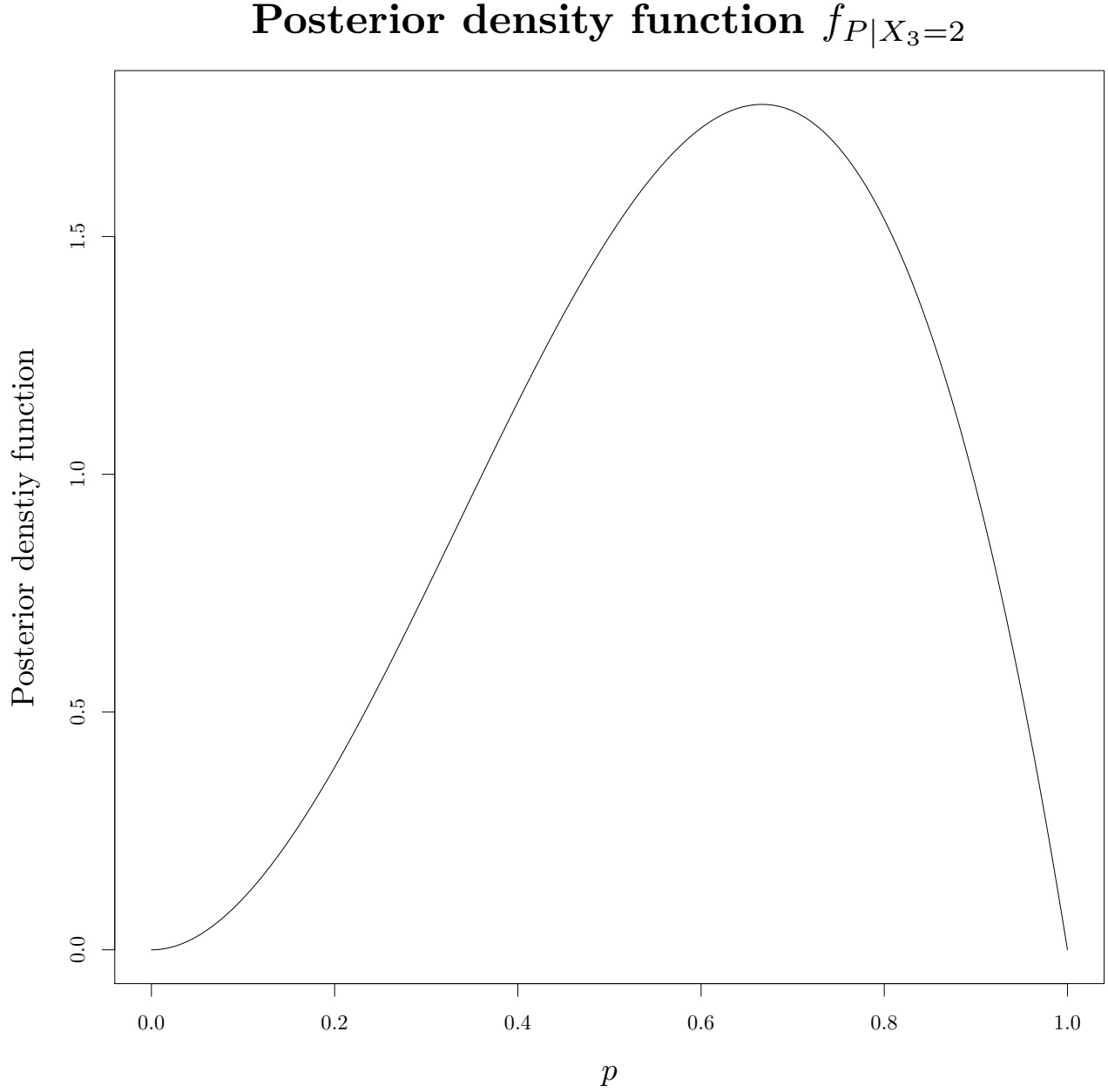
```
> # Take a sample from this discrete distribution
> samples = sample(p_grid, prob=posterior,
+                  size=m, replace=TRUE)
> samples
```

```
 [1] 0.5555556 0.6666667 0.5555556 0.5555556
 [5] 0.5555556 0.6666667 0.6666667 0.6666667
 [9] 0.2222222 0.7777778
```

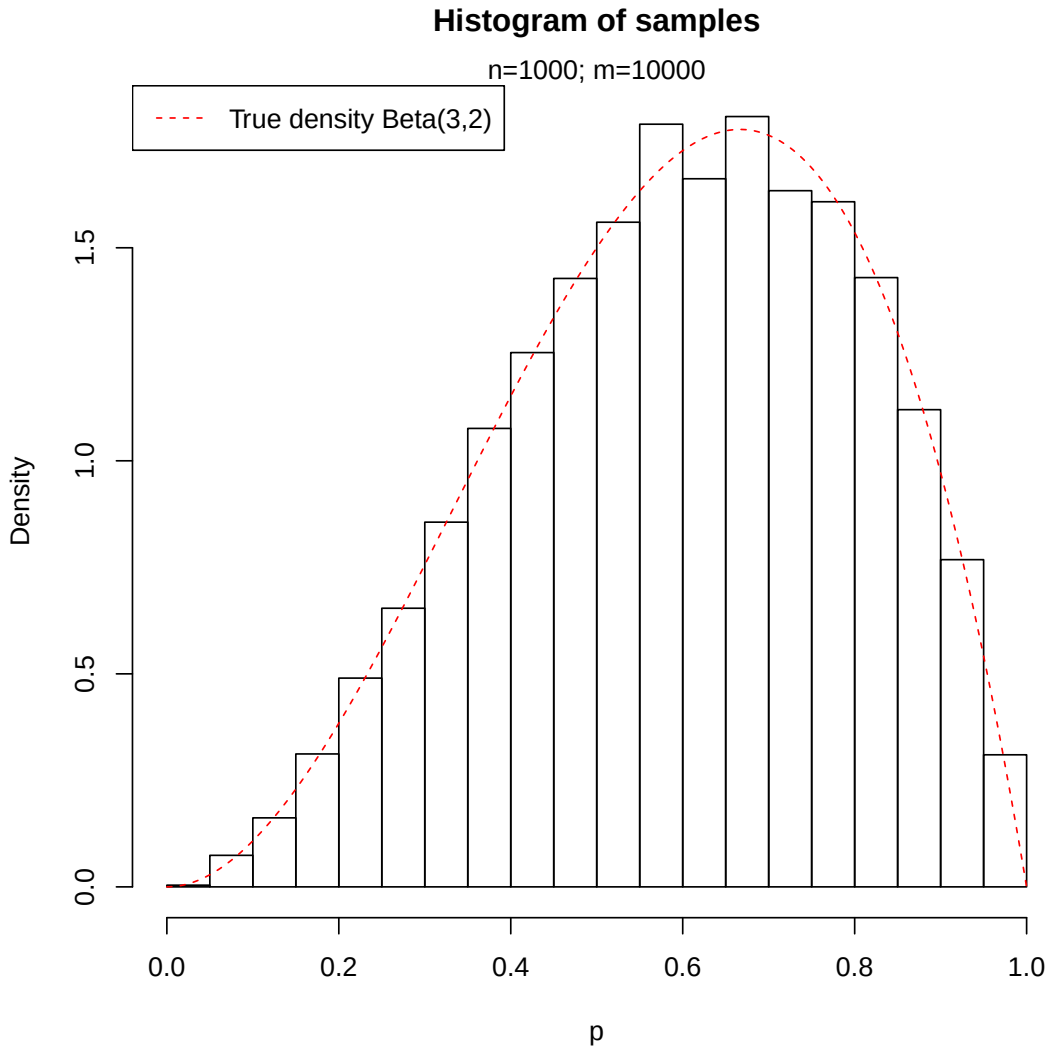
```
> p_grid
```

```
 [1] 0.0000000 0.1111111 0.2222222 0.3333333
 [5] 0.4444444 0.5555556 0.6666667 0.7777778
 [9] 0.8888889 1.0000000
```

- Note some values occurring more often than other values which reflects the posterior knowledge regarding p in a very rough sense.


Posterior density function $f_{P|X_3=2}$
Posterior denstiy function
0.0
0.5
1.0
1.5
0.0
0.2
0.4
0.6
0.8
1.0
p

- Of course, we should use more nodes $n = 1e3$ and a larger sample $m = 1e4$.



- A sample form a Monte Carlo simulation offers more than just a crude plot of the posterior density, which we already have before the simulation, i.e.

```
> # normalising constant
> A = (b-a) * sum(unstd.posterior) / n
```

- Having a sample means we can easily find an expectation of some function,

$$\mathbb{E}\left[g(y)\right] = \int_{\infty}^{\infty} g(y) f_{Y|X}(y \mid x)\, dy$$

```
> mean(samples)
```

```
[1] 0.5989582
```

```
> mean(log(samples/(1-samples))) # log odds/logit
```

```
[1] 0.4952244
```

- We could also obtain an estimate of the probability of p in certain interval

```
> sum(samples < 0.5) / m
```

```
[1] 0.3168
```

```
> sum(samples > 0.25 & samples < 0.75) / m
```

```
[1] 0.6899
```

- Of course, we could also reverse it

```
> median(samples); quantile(samples, 0.5)
```

```
[1] 0.6136136
      50%
0.6136136
```

```
> quantile(samples, c(0.025, 0.975))
```

```
     2.5%      97.5%
0.1941942 0.9309309
```

- Since we have discrete values, optimisation is easy

```
> p_grid[which.max(posterior)]
```

```
[1] 0.6666667
```

Q: How about the posterior predictive distribution?

$$f_{X_k^*|X_3=2} = \int_0^1 f_{X_k^*|P} \cdot f_{P|X_3=2}\, dp$$

$$= \frac{k^*!}{x_k^*!(k^*-x_k^*)!} \cdot \frac{\Gamma(\alpha^*+\beta^*)}{\Gamma(\alpha^*)\Gamma(\beta^*)} \cdot \frac{\Gamma(x_k^*+\alpha^*)\Gamma(k^*-x_k^*+\beta^*)}{\Gamma(k^*+\alpha^*+\beta^*)}$$

```
> ppsamples = double(m)
> for (i in 1:m){
+   p = samples[i]
+   ppsamples[i] = rbinom(1, size=3, p)
+ }
> head(ppsamples)
```

```
[1] 1 1 3 3 2 1
```

- Note the above example has only a single scalar unknown y,

$$f_{Y|X} \propto f_{X|Y} f_Y$$

it illustrates what Monte Carlo simulation can provide. It is chosen since it is simple/familiar, and knowing the exact posterior makes comparisons easy.

- In practice, Monte Carlo simulation is used for complicated Bayesian models

$$\begin{aligned} f_{\{\mathbf{P},\alpha,\beta\}|\mathbf{Y}} &\propto f_{\mathbf{Y}|\{\mathbf{P},\alpha,\beta\}} \cdot f_{\mathbf{P}|\{\alpha,\beta\}} \cdot f_{\alpha,\beta} \\ &= f_{\alpha,\beta} \prod_{j=1}^{71} \binom{k_j}{y_j} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} p_j^{y_j+\alpha-1} (1-p_j)^{k_j-y_j+\beta-1} \end{aligned}$$

with a large number of unknowns, however, what Monte Carlo can provide and how it can provide them remain largely the same.

Q: But do you notice any flaw with this simple sampling scheme?

- The above sampling scheme is based on a grid approximation, which means it will fail miserably in high dimensions, we must replace this direct sampling scheme before we can apply Monte Carlo in high dimensions.