Lab 8.5: Review for Exam II

Olivier Binette

Friday October 16, 2020

Agenda

- Annoucements
- Review of gaussian mixture models
- Appendix: general review of sampling methods

Announcements

Announcements

- Please vote!
  - Forgot to register? Not a problem! Bring an ID and proof of residence (e.g. bank statement or utility bill) to vote **before October 31** at an early voting site.
- Please fill out the TA section of your class evaluation!
  - Your evaluations are very important to me.

Announcements

- Please vote!
  - Forgot to register? Not a problem! Bring an ID and proof of residence (e.g. bank statement or utility bill) to vote **before October 31** at an early voting site.
- Please fill out the TA section of your class evaluation!
  - Your evaluations are very important to me.

Review of gaussian mixture models

Review of gaussian mixture models

Appendix: review of sampling methods

Review of sampling methods

1. Inverse CDF method
2. Rejection sampling
3. MCMC methods
   - Metropolis-Hastings
   - **Gibbs sampling**

1. Inverse CDF method

Goal: Generate samples $X_1, X_2, \ldots, X_n$ from a distribution on $\mathbb{R}$ with CDF F.

The trick: If F is invertible and $U \sim \text{Unif}(0, 1)$, then $X = F^{-1}(U)$ has the correct distribution.

When is it used? - Works only for *univariate* distributions. - You need to be able to evaluate F^{-1}.

1. Inverse CDF method

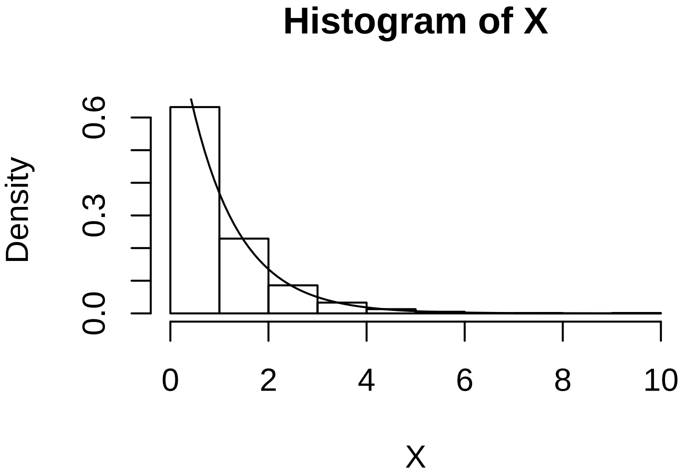
Example: Sampling from an Exp(λ) distribution

1. The CDF of $X \sim \text{Exp}(\lambda)$ is $F(x) = 1 - e{-}\lambda x$.
2. Its inverse is $F^{-1}(u) = -\log(1-u)/\lambda$.

```
F.inv <- function(u, lambda=1) -log(1-u)/lambda

n = 1000
X = F.inv(runif(n))
```

1. Inverse CDF method

```r
hist(X, prob=TRUE)
curve(dexp(x), add=TRUE)
```



2. Rejection sampling

Goal: Generate samples $X_1, X_2, \dots, X_n$ from a distribution with density (proportional to) $p(x)$.

The trick: Try to find a density $q(x)$ which you can sample from and such that $cq(x) \geq p(x)$ for some c.

Algorithm:

1. Generate $X \sim q(x)$ and $Y \sim \mathrm{Unif}(0, cq(X))$.
2. If $Y < p(X)$, then return X. Otherwise go back to step 1.

2. Rejection sampling

Goal: Generate samples $X_1, X_2, \ldots, X_n$ from a distribution with density (proportional to) $p(x)$.

The trick: Try to find a density $q(x)$ which you can sample from and such that $cq(x) \geq p(x)$ for some c.

Algorithm:

1. Generate $X \sim q(x)$ and $Y \sim \text{Unif}(0, cq(X))$.
2. If $Y < p(X)$, then return X. Otherwise go back to step 1.

2. Rejection sampling

Goal: Generate samples $X_1, X_2, \ldots, X_n$ from a distribution with density (proportional to) $p(x)$.

The trick: Try to find a density $q(x)$ which you can sample from and such that $cq(x) \geq p(x)$ for some c.

Algorithm:

1. Generate $X \sim q(x)$ and $Y \sim \text{Unif}(0, cq(X))$.
2. If $Y < p(X)$, then return X. Otherwise go back to step 1.

2. Rejection sampling

Example:

Let $p(x) = \sin^2(\pi x)$ be defined on $[0, 1]$ and let $q(x) = 1$ for all x. Take $c = 1$ since $p(x) \leq 1$.

```
p <- function(x) sin(pi*x)^2
q <- Vectorize(function(x) 1)

# Vectorized form of rejection sampling:
k = 5000
X = runif(k) # Samples from q
Y = runif(k) # Samples uniform between 0 and cq(X)
X = X[Y < p(X)] # Only keep the X for which Y < p(X).

length(X)/5000 # Acceptance rate
```

```
## [1] 0.4876
```

2. Rejection sampling

Example:

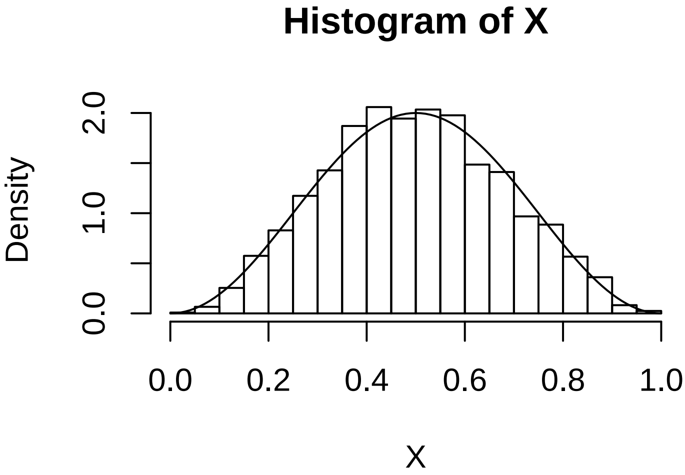
Let $p(x) = \sin^2(\pi x)$ be defined on $[0, 1]$ and let $q(x) = 1$ for all x. Take $c = 1$ since $p(x) \leq 1$.

```
p <- function(x) sin(pi*x)^2
q <- Vectorize(function(x) 1)

# Vectorized form of rejection sampling:
k = 5000
X = runif(k) # Samples from q
Y = runif(k) # Samples uniform between 0 and cq(X)
X = X[Y < p(X)] # Only keep the X for which Y < p(X).

length(X)/5000 # Acceptance rate
```

```
## [1] 0.4876
```

2. Rejection sampling

```r
hist(X, prob=TRUE, breaks=20)
curve(2*p(x), add=TRUE)
```



2. Rejection sampling

When is rejection sampling used?

- Works great for *univariate* densities (just like the inverse CDF method).
- You don't even need a normalizing constant for p (e.g. posterior distributions!).
- Trickier for higher-dimensional distributions (that's where Gibbs sampling comes in).

3. Metropolis-Hastings

Goal: Generate a Markov Chain $X^{(1)}, X^{(2)}, \ldots, X^{(n)}$ with stationary distribution (proportional to) $p(x)$.

- In practice the $X^{(s)}$ are seen as correlated samples from the density proportional to $p(x)$.

The trick:

- Given $X^{(s)} = x$, propose $X^{(s+1)} = x^*$ following some distribution $J(x^* \mid x)$.
- Accept the proposal with probability

$$\alpha = \min\left\{1, \frac{p(x^*)J(x \mid x^*)}{p(x)J(x^* \mid x)}\right\},$$

- Otherwise set $X^{(s+1)} = X^{(s)} = x$.

Metropolis-Hastings

https://gfycat.com/relievedglossyhowlermonkey

3. Metropolis-Hastings

When is it used?

- To sample from high-dimensional distributions
- No need to know a normalizing constant for $p(x)$ (e.g. posterior distributions!).

What to watch out for?

- Convergence issues: you want your samples to be a good approximation to p and to not be too correlated with one another.
- The acceptance rate of the proposals can help diagnose issues, but it doen't tell you about convergence.
- You need to look at convergence diagnostics.

4. Gibbs sampling

Goal: Generate a Markov Chain $X^{(1)}, X^{(2)}, \ldots, X^{(n)}$ with stationary distribution (proportional to) $p(x)$, where $x = (x_1, x_2, \ldots, x_k)$.

The trick: Reduce to sampling from the *full conditional distributions* $p(x_i \mid x_{(-i)})$.

Algorithm:

1. Initialize $X^{(1)} = (X_1^{(1)}, X_2^{(1)}, \ldots, X_k^{(1)})$ to fixed values.
2. For $s = 2, 3, \ldots, n$, do:

- $X_1^{(s)} \sim p(x_1 \mid X_2^{(s-1)}, X_2^{(s-1)}, \ldots, X_k^{(s-1)})$
- $X_2^{(s)} \sim p(x_2 \mid X_1^{(s)}, X_3^{(s-1)}, \ldots, X_k^{(s-1)})$
- $X_3^{(s)} \sim p(x_3 \mid X_1^{(s)}, X_2^{(s)}, X_4^{(s-1)}, \ldots, X_k^{(s-1)})$
- $\vdots$
- $X_k^{(s)} \sim p(x_k \mid X_1^{(s)}, X_2^{(s)}, \ldots, X_{k-1}^{(s)})$

4. Gibbs sampling

Example: Go back to the gaussian mixture model example.

When is it used?:

- To sample from high-dimensional distributions
- No need to know a normalizing constant for $p(x)$ (e.g. posterior distributions!).
- You need to derive the full-posterior distributions.

What to watch out for:

- Convergence issues: you want your samples to be a good approximation to p and to not be too correlated with one another.
- You need to look at convergence diagnostics.