LECTURE 21: MONTE CARLO METHODS

STAT 598z: Introduction to computing for statistics

Vinayak Rao

Department of Statistics, Purdue University

April 11, 2017

We want to calculate integrals/summations

- often expectations w.r.t. some probability distribution $p(x)$

$$\mu := \mathbb{E}_p[f] = \int_{\mathcal{X}} f(x)p(x)\mathrm{d}x$$

Monte Carlo integration

We want to calculate integrals/summations

- often expectations w.r.t. some probability distribution $p(x)$

$$\mu := \mathbb{E}_p[f] = \int_{\mathcal{X}} f(x)p(x)\mathrm{d}x$$

What is the prob. a game of patience (solitaire) is solvable?

$$P(\text{Solvable}) = \frac{1}{|\Pi|}\sum_{\Pi} \mathbb{1}(\Pi \textit{ is solvable})$$

Monte Carlo integration

We want to calculate integrals/summations

- often expectations w.r.t. some probability distribution $p(x)$

$$\mu := \mathbb{E}_p[f] = \int_{\mathcal{X}} f(x)p(x)\mathrm{d}x$$

If we drop 3 points on the plane, each Gaussian distributed, what is average the area of the resulting triangle?

$$\mathbb{E}[A] = \int A(x_1, x_2, x_3)p(x_1)p(x_2)p(x_3)\mathrm{d}x_1\mathrm{d}x_2\mathrm{d}x_3$$

Monte Carlo integration

We want to calculate integrals/summations

- often expectations w.r.t. some probability distribution $p(x)$

$$\mu := \mathbb{E}_p[f] = \int_{\mathcal{X}} f(x)p(x)\mathrm{d}x$$

For dataset (X, y), what is is average loss if you randomly choose a weight-vector according to some distribution (e.g. `rnorm`)?

$$\mathbb{E}_w[\mathcal{L}(X, y)] = \int (y - w^T X)^2 p(w)\mathrm{d}w$$

Monte Carlo integration

We want to calculate integrals:

$$\mu := \mathbb{E}_p[f] = \int_{\mathcal{X}} f(x)p(x)\mathrm{d}x$$

Monte Carlo integration

We want to calculate integrals:

$$\mu := \mathbb{E}_p[f] = \int_{\mathcal{X}} f(x)p(x)\mathrm{d}x$$

Sampling approximation: rather than visit all points in $\mathcal{X}$, calculate a summation over a finite set.

$$\hat{\mu} \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

Monte Carlo integration

We want to calculate integrals:

$$\mu := \mathbb{E}_p[f] = \int_{\mathcal{X}} f(x)p(x)\mathrm{d}x$$

Sampling approximation: rather than visit all points in $\mathcal{X}$, calculate a summation over a finite set.

$$\hat{\mu} \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

Monte Carlo approximation:

- Obtain point by sampling from $p(x)$

$$x_i \sim p$$

Is this a good idea?

- Very simple
- Unbiased

Monte Carlo integration

Is this a good idea?

- Very simple
- Unbiased

If $x_i \sim p$,

$$\mathbb{E}_p[\hat{\mu}] = \mathbb{E}_p\left[\frac{1}{N}\sum_{i=1}^{N} f(x)\right] = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_p[f] = \mu \qquad \text{Unbiased estimate}$$

Monte Carlo integration

Is this a good idea?

- Very simple
- Unbiased

If $x_i \sim p$,

$$\mathbb{E}_p[\hat{\mu}] = \mathbb{E}_p\left[\frac{1}{N}\sum_{i=1}^{N} f(x)\right] = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_p[f] = \mu \qquad \text{Unbiased estimate}$$

$$\mathrm{Var}_p[\hat{\mu}] = \frac{1}{N}\mathrm{Var}_p[f], \qquad \text{Error = StdDev} \propto N^{-1/2}$$

Monte Carlo integration

Is this a good idea?

- In low-dims, use numerical methods like quadrature
- In high-dims, numerical methods become infeasible

Monte Carlo integration

Is this a good idea?

- In low-dims, use numerical methods like quadrature
- In high-dims, numerical methods become infeasible
  E.g. Simpson's rule in d-dimensions, with N grid points:

$$\text{error} \propto N^{-4/d}$$

Monte Carlo integration

Is this a good idea?

- In low-dims, use numerical methods like quadrature
- In high-dims, numerical methods become infeasible
  E.g. Simpson's rule in d-dimensions, with N grid points:

$$\text{error} \propto N^{-4/d}$$

  Monte Carlo integration:

$$\text{error} \propto N^{-1/2}$$

Monte Carlo integration

Is this a good idea?

- In low-dims, use numerical methods like quadrature
- In high-dims, numerical methods become infeasible
  E.g. Simpson's rule in d-dimensions, with N grid points:

$$\text{error} \propto N^{-4/d}$$

Monte Carlo integration:

$$\text{error} \propto N^{-1/2}$$

Independent of dimensionality!

Generating random variables

- The simplest useful probability distribution $\texttt{Unif}(0, 1)$.
- In theory, can be used to generate any other RV.
- Easy to generate uniform RVs on a deterministic computer?

Generating random variables

- The simplest useful probability distribution $\texttt{Unif}(0, 1)$.
- In theory, can be used to generate any other RV.
- Easy to generate uniform RVs on a deterministic computer?

No!

Generating random variables

- The simplest useful probability distribution $\texttt{Unif}(0, 1)$.
- In theory, can be used to generate any other RV.
- Easy to generate uniform RVs on a deterministic computer?

No!

- Instead: *pseudorandom* numbers.

Generating random variables

- The simplest useful probability distribution $\texttt{Unif}(0, 1)$.
- In theory, can be used to generate any other RV.
- Easy to generate uniform RVs on a deterministic computer?

No!

- Instead: *pseudorandom* numbers.
- Map a seed to a 'random-looking' sequence.

Generating random variables

- The simplest useful probability distribution `Unif`$(0, 1)$.
- In theory, can be used to generate any other RV.
- Easy to generate uniform RVs on a deterministic computer?

No!

- Instead: *pseudorandom* numbers.
- Map a seed to a 'random-looking' sequence.
- Downside: `http://boallen.com/random-numbers.html`

Generating random variables

- The simplest useful probability distribution $\texttt{Unif}(0, 1)$.
- In theory, can be used to generate any other RV.
- Easy to generate uniform RVs on a deterministic computer?

No!

- Instead: *pseudorandom* numbers.
- Map a seed to a 'random-looking' sequence.
- Downside: `http://boallen.com/random-numbers.html`
- Upside: Can use seeds for reproducibility or debugging

```
set.seed(1)
```

Generating random variables

- The simplest useful probability distribution $\texttt{Unif}(0, 1)$.
- In theory, can be used to generate any other RV.
- Easy to generate uniform RVs on a deterministic computer?

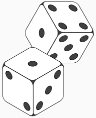
No!

- Instead: *pseudorandom* numbers.
- Map a seed to a 'random-looking' sequence.
- Downside: `http://boallen.com/random-numbers.html`
- Upside: Can use seeds for reproducibility or debugging

```
set.seed(1)
```

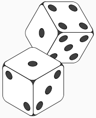
- Careful with batch/parallel processing.

Example Monte Carlo sampling



x, y are a pair of dice. What is $p(x, y)$?

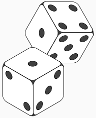
Example Monte Carlo sampling



x, y are a pair of dice. What is $p(x, y)$?

What is $\mathbb{E}[\min(x, y)] = \sum_{i=1}^{6} \sum_{j=1}^{6} \min(x, y) p(x, y)$

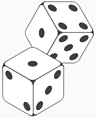
Example Monte Carlo sampling



x, y are a pair of dice. What is $p(x, y)$?

What is $\mathbb{E}[\min(x, y)] = \sum_{i=1}^{6} \sum_{j=1}^{6} \min(x, y) p(x, y)$

Don't really need Monte Carlo here (but what if we had a 100 dice?), but let's try it anyway. How?



x, y are a pair of dice. What is $p(x, y)$?

What is $\mathbb{E}[\min(x, y)] = \sum_{i=1}^{6} \sum_{j=1}^{6} \min(x, y) p(x, y)$

Don't really need Monte Carlo here (but what if we had a 100 dice?), but let's try it anyway. How?

Roll a pair of dice N times. Call the ith outcome (x_i, y_i). Then

$$\mathbb{E}[\min(x, y)] \approx \frac{1}{N} \sum_{i=1}^{N} \min(x_i, y_i)$$

Generating random variables

In `R` we generate uniform random variables via `runif`

Additional functions include

`rnorm()`, `rgamma()`, `rexp()`, `sample()` etc.

Generating random variables

In `R` we generate uniform random variables via `runif`

Additional functions include

`rnorm()`, `rgamma()`, `rexp()`, `sample()` etc.

In theory, we can generate any other random variable by transforming a uniform (or any other) random variable:

$$u \sim \texttt{Unif}(0,1), \quad x = f(u)$$

In practice, finding this f is too hard. Need other approaches.

Let $X = (x_1, \ldots, x_{100})$ be a hundred dice.
What is $p(\sum_{d=1}^{100} x_d \geq 450)$?

Rare event simulation:

Let $X = (x_1, \ldots, x_{100})$ be a hundred dice.
What is $p(\mathrm{Sum}(X) \geq 450)$?
(where $\mathrm{Sum}(X) = \sum_{d=1}^{100} x_d$)

Let $X = (x_1, \ldots, x_{100})$ be a hundred dice.
What is $p(\mathrm{Sum}(X) \geq 450)$?
(where $\mathrm{Sum}(X) = \sum_{d=1}^{100} x_d$)

$$
\begin{aligned}
p(\mathrm{Sum}(X) \geq 450) &= \sum \delta(\mathrm{Sum}(X) \geq 450) p(X) \\
&= \mathbb{E}_p[\delta(\mathrm{Sum}(X) \geq 450)]
\end{aligned}
$$

- $\delta(\cdot)$ is the indicator function
- $\delta(\mathit{condition}) = 1$ if *condition* is true, else 0.

Naive Monte Carlo sampling

- Propose from $p(x)$
- Calculate $\frac{1}{N} \sum_{i=1}^{N} \delta(\mathrm{Sum}(X_i))$

Naive Monte Carlo sampling

- Propose from $p(x)$
- Calculate $\frac{1}{N}\sum_{i=1}^{N}\delta(\mathrm{Sum}(X_i))$

Most $\delta(\mathrm{Sum}(X_i))$ terms will be 0

High variance

Importance sampling: assigns importance weights to samples.

Importance sampling

Importance sampling: assigns importance weights to samples.

Scheme:

- Draw a proposal x_i from $q(\cdot)$
- Assign it a weight $w_i = p(x_i)/q(x_i)$

Importance sampling

Importance sampling: assigns importance weights to samples.

Scheme:

- Draw a proposal x_i from $q(\cdot)$
- Assign it a weight $w_i = p(x_i)/q(x_i)$

$$\mu = \int_{\mathcal{X}} f(x)p(x)\mathrm{d}x \approx \frac{1}{N}\sum_{i=1}^{N} w_i f(x_i) := \mu_{imp}$$

Importance sampling: assigns importance weights to samples.

Scheme:

- Draw a proposal x_i from $q(\cdot)$
- Assign it a weight $w_i = p(x_i)/q(x_i)$

$$\mu = \int_{\mathcal{X}} f(x)p(x)\mathrm{d}x \approx \frac{1}{N}\sum_{i=1}^{N} w_i f(x_i) := \mu_{imp}$$

$$\mathbb{E}[\mu_{imp}] = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}[w_i f(x_i)] = \int_{\mathcal{X}} \frac{p(x)}{q(x)} f(x)q(x)\mathrm{d}x$$

Importance sampling

Importance sampling: assigns importance weights to samples.

Scheme:

- Draw a proposal x_i from $q(\cdot)$
- Assign it a weight $w_i = p(x_i)/q(x_i)$

$$\mu = \int_{\mathcal{X}} f(x)p(x)\mathrm{d}x \approx \frac{1}{N}\sum_{i=1}^{N} w_i f(x_i) := \mu_{imp}$$

$$\mathbb{E}[\mu_{imp}] = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}[w_i f(x_i)] = \int_{\mathcal{X}} \frac{p(x)}{q(x)} f(x)q(x)\mathrm{d}x$$

Importance Sampling (contd)

When does this make sense?

Sometimes it's easier to simulate from $q(x)$ than $p(x)$

Importance Sampling (contd)

When does this make sense?

Sometimes it's easier to simulate from $q(x)$ than $p(x)$

Sometimes it's better to simulate from $q(x)$ than $p(x)$!

To reduce variance. E.g. rare event simulation.

For 100 dice, what is $p(\text{Sum} > 450)$? A better choice might be to bias the dice.

E.g. $\quad q(x_i = v) \propto v \quad (\text{for } v \in \{1, \ldots 6\})$

For 100 dice, what is $p(\text{Sum} > 450)$? A better choice might be to bias the dice.
E.g. $\quad q(x_i = v) \propto v \quad (\text{for } v \in \{1, \ldots 6\})$

- Propose from $q(x)$
- Calculate weights $w(X_i) = p(X_i)/q(X_i)$
- Calculate $\frac{1}{N}\sum_{i=1}^{N} w(X_i)\delta(\text{Sum}(X_i))$

Importance sampling

For 100 dice, what is $p(\text{Sum} > 450)$? A better choice might be to bias the dice.
E.g. $\quad q(x_i = v) \propto v \quad$ (for $v \in \{1, \ldots 6\}$)

- Propose from $q(x)$
- Calculate weights $w(X_i) = p(X_i)/q(X_i)$
- Calculate $\frac{1}{N}\sum_{i=1}^{N} w(X_i)\delta(\text{Sum}(X_i))$

Gives a better estimate of
$p(\text{Sum}(X) \geq 500) = \sum \delta(\text{Sum}(X) \geq 500)p(X)$