MCMC: Markov Processes, Chains

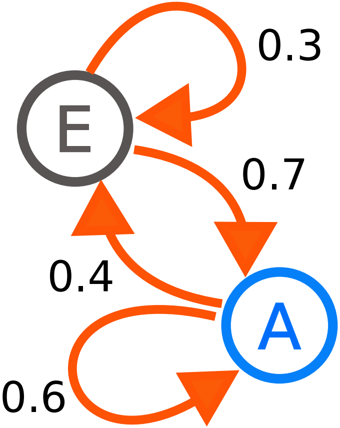
DRME

2024 01 20

- A **Markov process** exhibits the *Markov property*.
- Predictions about future outcomes depend *solely on the current state*.
- *Independent of the process's history*.

- A **Markov chain** is a specific type of Markov process.
- It has a *discrete state space or index set*.
- Can be defined with *discrete or continuous time* and a *countable state space*.

Wikipedia:Markov



Markov Chains in R

R Code: Simulating a Simple Markov Chain

```
# Define transition matrix
transition_matrix <- matrix(c(0.7, 0.3, 0.2, 0.8),
                            nrow = 2, byrow = TRUE)

# Initial state
initial_state <- c(0.5, 0.5)

# Simulate Markov Chain
set.seed(123)
num_steps <- 10
states <- matrix(0, nrow = num_steps, ncol = 2)

for (i in 1:num_steps) {
  if (i == 1) states[i, ] <- initial_state
  else states[i, ] <- states[i - 1, ] %*% transition_matrix
}
```

Applications of Markov Chains

- Markov chains are used to model sequences of events.
- Probability of each event depends *only on the previous state*.
- Applications in various fields:
  - Cruise control systems
  - Customer queues
  - Currency exchange rates
  - Animal population dynamics

Introduction to MCMC Methods

- MCMC methods are statistical algorithms for sampling from probability distributions.
- They create a Markov chain with the desired distribution as its equilibrium state.
- Samples are generated by recording states from the chain.
- Accuracy improves with more steps.
- Various algorithms, including Metropolis–Hastings, are used in constructing these chains.

MCMC for Generating Samples

- MCMC methods generate samples from a continuous random variable.
- Samples are proportional to a known function's probability density.
- Used for evaluating integrals over the variable, e.g., expected value or variance.

Ensembles of Chains

- Ensembles of chains are developed by initiating stochastic processes or ”walkers” from distant points.
- Walkers move randomly based on an algorithm prioritizing areas with higher contributions to the integral.
- Higher probabilities are assigned to these areas.

Curse of Dimensionality

- Despite their effectiveness in multi-dimensional problems, MCMC methods face the curse of dimensionality.
- High-dimensional spaces cause regions of higher probability to stretch and get lost in vast spaces with little contribution to the integral.
- Methods like reducing walker step size can be employed, but it leads to high autocorrelation and increased computational expense.

Advanced Methods

- To overcome challenges, advanced methods like Hamiltonian Monte Carlo and the Wang and Landau algorithm have been developed.
- These methods use techniques to reduce autocorrelation while keeping the process in integral-contributing regions.
- Although they rely on intricate theories and are harder to implement, they often converge faster than simpler approaches.

Metropolis–Hastings Algorithm

- Generates a Markov chain using a proposal density for new steps.
- Includes a mechanism for rejecting certain proposed moves.
- Serves as a general framework, encompassing the original Metropolis algorithm and subsequent alternatives.

Gibbs Sampling

- Designed for multi-dimensional target distributions.
- Updates each coordinate based on its full conditional distribution given other coordinates.
- Special case of Metropolis–Hastings with a uniform acceptance rate of 1; does not require tuning.
- Commonly used, structure resembles coordinate ascent variational inference.

Metropolis-Adjusted Langevin Algorithm (MALA) and Gradient-Based Methods

- MALA and similar methods use the gradient (and possibly second derivative) of the log target density.
- Propose steps likely to move in the direction of higher probability density.

Pseudo-Marginal Metropolis–Hastings

- Replaces direct evaluation of the target distribution density with an unbiased estimate.
- Useful when the target density is not analytically available (e.g., in latent variable models).

Slice Sampling

- Involves sampling from a distribution by alternating between uniform sampling in the vertical direction and uniform sampling from the horizontal 'slice.'
- Based on the principle of sampling uniformly from the region under the plot of the density function.

Introduction to Metropolis–Hastings Algorithm

- Draws samples from any probability distribution with probability density $P(x)$.
- Requires a function $f(x)$ proportional to $P(x)$ with calculable values.
- Overcomes the challenge of computing the normalization factor in practice.

Algorithm Overview

- Generates a sequence of sample values.
- Distribution of values progressively approximates the desired distribution.
- Operates iteratively, forming a Markov chain.

Markov Chain Structure

- Next sample's distribution depends solely on the current sample.
- Iterative generation creates a Markov chain.

Iteration Process

- Proposes a candidate for the next sample based on the current sample.
- Accepts or rejects the candidate with a certain probability.
- Probability of acceptance determined by comparing $f(x)$ values for current and candidate samples with respect to the desired distribution.

Metropolis Algorithm Overview

- Generates samples from a probability distribution with density $P(x)$.
- Utilizes a function $f(x)$ proportional to $P(x)$ for the Markov Chain Monte Carlo (MCMC) method.
- Operates iteratively, attempting random moves in the sample space.

Initialization

- Choose an arbitrary starting point x_t.
- Select a symmetric proposal density function $g(x'|y)$, often a Gaussian distribution centered at y.

Iteration Process

- Generate a candidate x' from $g(x'|x_t)$.
- Calculate acceptance ratio $\alpha = P(x')/P(x_t)$.
- Accept or reject the candidate based on u, a uniform random number.

Acceptance or Rejection

- If $u \leq \alpha$, accept the candidate ($x_{t+1} = x'$).
- If $u > \alpha$, reject the candidate ($x_{t+1} = x_t$).

Intuition behind Acceptance

- $\alpha > 1$: Move is always accepted for more probable points.
- $\alpha \leq 1$: Move is occasionally rejected for less probable points.

Algorithm Characteristics

- Tends to stay in high-density regions of $P(x)$.
- Occasionally explores low-density regions.
- Effectively generates samples following the desired distribution.

Introduction to Gibbs Sampling

- Basic version as a special case of the Metropolis–Hastings algorithm.
- Extended versions serve as a general framework for sampling from a set of variables.
- Integration with Metropolis–Hastings or slice sampling for enhanced flexibility.

Applicability of Gibbs Sampling

- Suitable when the joint distribution is not explicitly known or challenging to sample directly.
- Assumes known conditional distributions of variables, making sampling feasible.
- Useful for sampling from a large set of variables sequentially.

Gibbs Sampling Algorithm

- Generates samples from each variable in turn, conditioned on the current values of other variables.
- Sequential process to sample each variable or group of variables.
- The sequence of samples forms a Markov chain.

Gibbs: Sampling Procedure algorithm

1. Initialization:
   - Begin with an initial value $\mathbf{X}^{(0)}$.
2. Iterative Sampling:
   - Obtain k samples of $\mathbf{X}$ through sequential conditional sampling.
   - Update each component based on conditional distributions.
   - Use values from the previous sample for certain components.
3. Repeat:
   - Iterate the above step k times.