Sampling

In this chapter we will explore the generation of random numbers and techniques to sample from probability distribution functions

Random numbers

In order to obtain random samples from a probability distribution function we have to have a way to create random numbers. In principle, you could throw dice and then type the numbers into the computer, but that seems quite tedious. Instead, a lot of effort has been spent on developing algorithms to create so-called pseudo-random number generators. We will begin by looking at a particular instantiation of random number generators that are called "Linear congruential generator"

$$X_{n+1} = (aX_n + c) \mod m$$

even though these number generators are not much used anymore, they illustrate nicely the basic principle of a pseudo-random number generator. For a given u,a and m, we can create a string of random numbers once we pick a so-called "seed" number X_1. The parameters u,a, and m are chosen in such a way that the numbers do not repeat in within a particular range and that consequitve numbers do not exhibit correlations (for example Numerical Recipies suggests $m = 2^{32}$, $a = 1664525$, $c = 1013904223$). Given that they come from a deterministic alogrithm this is impossible to achieve, but modern random number generators are pretty close to random for most practical purposes. It is therefore a good idea to test your results using several independent runs of random numbers to make sure that the result does not depend on the random "seed".

Sampling from distributions

In python, we have access to many forms of random numbers and distributions, but the most common random number generator produces a random float from a uniform distribution covering the interval $[0, 1)$ and can be accessed using the function np.random.rand()

In last weeks homework we explored how to use such random numbers to obtain a sample from a normal distribution.

Sampling from any pdf

Now we want to explore the question of how to obtain a random sample from an arbitrary probability function. The most general solution to this problem is to use the inverse cummulative probability density function (cdf). The basic idea is that cdf is given by the integral of a probability function $pdf(\xi)$ from $-\inf$ to x:

$$cdf(x) = \int_{-\inf}^{x} pdf(\xi) d\xi$$

It is clear that since the $pdf(\xi)$ is normalized, the $cdf(x)$ will have a range from $[0, 1]$. This immediately suggests the following strategy: If we can find the inverse of the cdf then we can map the range of $[0, 1]$ to all the accessible values of the pdf. So in order to create a random sample of the pdf we feed a stream of random numbers in the range $[0, 1]$ into the inverse cdf function. The only technical problem is to properly integrate the pdf and to inverse the resulting cdf. We will explore this technique in the Bioassay example.

Markov-Chain Monte-Carlo

Bayes posterior is proportional to

$$\pi(x) = P(D \mid x)P(x)$$

Markov-Chain Monte-Carlo is capable of drawing a sample $\{x_0, x_1, x_2, \ldots\}$ without the need for normalization.

The strategy is to sample a Markov chain $\{x_0, x_1, x_2, \ldots\}$ where each point is stochastically determined by the previous one by the distribution $p(x_i \mid x_{i-1})$. Even though this chain will be correlated (since it depends on the previous point drawn), $p(x_i \mid x_{i-1})$ can be designed to make the sequence ergodic meaning that it will visit every point x in proportion to $\pi(x)$

Any distribution $p(x_i \mid x_{i-1})$ that obeys "detailed balance" results in an ergodic sequence:

$$\pi(x_1)p(x_2 \mid x_1) = \pi(x_2)p(x_1 \mid x_2)$$

This can be proven in the following way:

$$\int \pi(x_1)p(x_2 \mid x_1)dx_1 = \pi(x_2) \int p(x_1 \mid x_2)dx1 = \pi(x_2)$$

This does not solve the problem though of how to choose $p(x_i \mid x_{i-1})$.

The algorithm below is called the "Metropolis-Hastings" algorithm and is based on papers by Metropolis, Rosenbluth, Rosenbluth, Teller and Teller (1953) and Hastings (1970):

First we pick a proposal distribution $q(x_2 \mid x1)$ (for example a multivariate normal centered around x_1 as a typical example).

Then:

1. Generate a candidate point x_{2c} that is drawn from the proposal distribution
2. Calculate the acceptance probability
$$\alpha(x_1, x_{2c}) = \min\Big(1, \frac{\pi(x_{2c})q(x_1 \mid x_{2c})}{\pi(x_1)q(x_{2c} \mid x_1)}\Big)$$
if $q(x_2 \mid x1)$ is symmetric it cancels
3. Choose $x_2 = x_{2c}$ with probability α, and $x_2 = x_1$ with $1 - \alpha$ for $(x_2 \neq x_1)$:
$$p(x_2 \mid x1) = q(x_2 \mid x1)\alpha(x_1, x_2)$$

Proof:

$$\alpha(x_1, x_{2c}) = \min\Big(1, \frac{\pi(x_{2c})q(x_1 \mid x_{2c})}{\pi(x_1)q(x_{2c} \mid x_1)}\Big)$$

$$\begin{aligned}\pi(x_1)q(x_2 \mid x1)\alpha(x_1, x_2) &= \min[\pi(x_1)q(x_2 \mid x1), \pi(x_2)q(x_1 \mid x_2)]\\ &= \min[\pi(x_2)q(x_1 \mid x2), \pi(x_1)q(x_2 \mid x_1)]\\ &= \pi(x_2)q(x_1 \mid x2)\alpha(x_2, x_1)\end{aligned}$$

but

$$p(x_2 \mid x1) = q(x_2 \mid x1)\alpha(x_1, x_2)$$

Therefore:

$$\pi(x_1)p(x_2 \mid x_1) = \pi(x_2)p(x_1 \mid x_2)$$

which is the condition of "detailed balance"

Metropolis sampler

The original paper by Metropolis (1953) describes a simpler acceptance probability if the proposal probability is symmetric (meaning that the forward and backwards probabilities are equal).

$$\alpha(x_1, x_{2c}) = \min\Big(1, \frac{\pi(x_{2c})}{\pi(x_1)}\Big)$$

A good choice for the proposal pdf is a multivariate Normal distribution (with as many components as variables of π:

$$q(x_{2c}|x_1, \Sigma) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\Sigma|^{1/2}}exp\{-\frac{1}{2}(x_{2c} - x_1)^T\Sigma^{-1}(x_{2c} - x_1)\}$$

where D is the dimensionality of the variable x and Σ is the covariance matrix. Usually this matrix is chosen to only have diagonal elements with values σ^2.

The algorithm becomes the following:

1) pick a new x_{2c} using $q(x_{2_c} \mid x1)$

2) calculate acceptance probability:

$$\alpha(x_1, x_{2c}) = \min\left(1, \frac{\pi(x_{2c})}{\pi(x_1)}\right)$$

3) Accept x_{2c} with acceptance probability α, otherwise add old point x_1 to Markov-chain

Gibbs sampler

The Gibbs sampler is a special case of the Metropolis-Hastings sampling method.

To understand this sampler we have to define the "fully conditional distribution" of $\pi(x)$. It is the normalized distribution obtained by sampling along one coordinate direction while keeping all the other parameters fixed. We write this as $\pi(x \mid x^-)$

$$\alpha(x_1, x_{2c}) = \min\left(1, \frac{\pi(x_{2c} \mid x^-) q(x_1 \mid x_{2c}, x^-)}{\pi(x_1 \mid x^-) q(x_{2c} \mid x_1, x^-)}\right)$$

If we chose the proposal as $q(x_2 \mid x_1, x^-) = \pi(x_2 \mid x^-)$ then we always accept the proposal.

In practice Gibbs sampling looks like this:

1. Cycle through the coordinates of x
2. Hold all the other coordinates fixed
3. Sample from the one dimensional distribution along the non-fixed coordinate. This requires normalization, but only in one dimension.

In [ ]:

```

```