Gradient based MH sampling

This is an asymmetric MH sampling method. Specifically, we study Metropolis-adjusted Langevin algorithm (MALA) or Langevin Monte Carlo (LMC).

This MH sampling method again requires no mathematical derivations on the distributions like usual MH. You just need to be able to write down the likelihood and its derivatives.

In a multivariate setting, usual MH sampling recommends sampling each variable one-by-one. This may be computationally prohibitive. In that case, gradient based MH sampling like MALA allows to sample all the variables together.

It is increasingly similar to optimization methods.
Tips for applying gradient based MH:

- Since gradient of the log-likelihood can be both positive or negative, it is advisable to make the MH move carefully. You may need to apply the sampling on a transformed variable.
- The optimal acceptance rate (55%) is slightly higher than random walk MH. Usually between 0.45-0.7 produce good results. (The optimal acceptance rate result is based on Gaussian-type distributions, not generalizable, hence it is recommended to consider the range instead.)

The candidate update will, in this case, depend on the gradient of the log-likelihood. Specifically $\mathbf{x}_c = \mathbf{x}^{(t)} + \frac{\epsilon}{2}\frac{\partial \log P(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}=\mathbf{x}^{(t)}} + \delta$, where $\delta \sim \text{Normal}(0, \epsilon)$. [Unlike RW-MH where the update was $\mathbf{x}_c = \mathbf{x}^{(t)} + \delta$, where $\delta \sim \text{Normal}(0, \epsilon)$]

The contribution of the 'extra' term $\frac{\epsilon}{2}\frac{\partial \log P(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}=\mathbf{x}^{(t)}}$ is significant when the current value is away from the maxima of $P(\mathbf{x})$ and it is small near the maxima. This is because near maxima we should have $\frac{\partial \log P(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}=\mathbf{x}^{(t)}} \approx 0$

This proposal is accepted or rejected according to the Metropolis-Hastings algorithm, with acceptance probability

$$\alpha = \min\left\{1, \frac{P(\mathbf{x}_c)q(\mathbf{x}^{(t)} \mid \mathbf{x}_c)}{P(\mathbf{x}^{(t)})q(\mathbf{x}_c \mid \mathbf{x}^{(t)})}\right\},$$

where

$$q(\mathbf{x}' \mid \mathbf{x}) \propto \exp\left(-\frac{1}{2\epsilon}\|\mathbf{x}' - \mathbf{x} - \frac{\epsilon}{2}\nabla \log P(x)\|_2^2\right), \text{ the Gaussian density with variance } \epsilon$$

Why? $q(\mathbf{x}_c \mid \mathbf{x}^{(t)})$ is the 'transition' probability to move to $\mathbf{x}_c$ from $\mathbf{x}^{(t)}$. Based on the update equation, the noise δ we need to add is $\mathbf{x}_c - \mathbf{x}^{(t)} - \frac{\epsilon}{2}\frac{\partial \log P(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}=\mathbf{x}^{(t)}}$ and thus $P(\delta = \mathbf{x}_c - \mathbf{x}^{(t)} - \frac{\epsilon}{2}\frac{\partial \log P(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}=\mathbf{x}^{(t)}})$. Similarly $q(\mathbf{x}^{(t)} \mid \mathbf{x}_c) = P(\delta = \mathbf{x}^{(t)} - \mathbf{x}_c - \frac{\epsilon}{2}\frac{\partial \log P(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}=\mathbf{x}_c})$

as the noise we need to add to move to $\mathbf{x}^{(t)}$ from $\mathbf{x}_c$ is $\mathbf{x}^{(t)} - \mathbf{x}_c - \frac{\epsilon}{2}\frac{\partial \log P(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}=\mathbf{x}_c}$ based on the update equation. And we have $\delta \sim \text{Normal}(0, \epsilon)$.

This updating step is similar to Gradient ascent, where we try to maximize the objective function.

We want to sample from a multivariate t-distribution whose density is,

$$f(\mathbf{x}) = \frac{\Gamma\left[(\nu+p)/2\right]}{\Gamma(\nu/2)\nu^{p/2}\pi^{p/2}\left|\boldsymbol{\Sigma}\right|^{1/2}}\left[1+\frac{1}{\nu}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]^{-(\nu+p)/2}$$

with three parameters $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_p]^T$ location (real $p \times 1$ vector) $\boldsymbol{\Sigma}$ scale matrix (positive-definite real $p \times p$ matrix) ν is the degrees of freedom.

The statistical properties are:
Mean = $\boldsymbol{\mu}$
Variance=$\frac{\nu}{\nu-2}\boldsymbol{\Sigma}$ if $\nu > 2$. Or else undefined.

We set $p = 3$ i.e. 3-dimensional t-distribution and also $\nu = 30$.

Although the density looks daunting, gradient-based MH can easily from this distribution.

```
mu <- c(0, 0, 0)
nu <- 30
p <- 3
Sigma = matrix(c(1,0.5,0,
                 0.5,1,0.3,
                 0,0.3,1), 3, 3, byrow = T)

Sigmainv <- solve(Sigma)

sam <- LaplacesDemon::rmvt(n=1000, mu=mu, S=Sigma, df=3)

seed=1
Total_itr <- 10000
burn <- 5000
set.seed(seed)
#Initialization
x <- rnorm(3)
#log-likelihood of the data excluding the part not involving the data
llhood <- -((nu+p)/2)*log(1+sum((x-mu)*(Sigmainv%*%(x-mu)))/nu)
derivx <- function(x){ #derivative w.r.t. x
  -((nu+p)/2)*(1/(1+sum((x-mu)*(Sigmainv%*%(x-mu)))/nu))*(Sigmainv%*%(x-mu))/nu
}
```

```
itr <- 0
#Define lists to store postburn samples
xsam <- matrix(0, Total_itr, 3)
epsilon <- 1e-1
flag <- 0
while(itr < Total_itr){
  itr <- itr + 1
  #The update is like 'gradient-based update from the current value + some noise'
  temp <- x + derivx(x) * epsilon/2 + rnorm(3, sd = sqrt(epsilon))
  #If epsilon is small, then xc and x are very close => acceptance prob close to 1
  #=> force to increase acceptance
  #If epsilon is large, then xc and x will be far
  #=> acceptance prob will likely to reduce => force to decrease acceptance
  xc <- temp
  #log-likelihood of the candidate excluding the part not involving the candidate
  llhoodc <- -((nu+p)/2)*log(1+sum((xc-mu)*(Sigmainv%*%(xc-mu)))/nu)
  #log-likelihood of the current value excluding the part not involving the current value
  llhood <- -((nu+p)/2)*log(1+sum((x-mu)*(Sigmainv%*%(x-mu)))/nu)
  #log{q(current value|candidate)}
  q1 <- -sum((x-derivx(temp) * epsilon/2- temp)^2/epsilon) /2
  #log{q(candidate|current value)}
  q2 <- -sum((temp-derivx(x) * epsilon/2- x)^2/epsilon) /2
  #D = log(alpha)
  D <- llhoodc - llhood + q1 - q2
  u <- runif(1)
  #likelihood at candidate > likelihood at current value =>
  #exp(D) > 1 => D > 0

  #When you generate u as runif(1), 'u' bounded [0,1]. log(u) < 0 for all possibilities of u.
  #log(u) < D for any 'u', generated.
  #D is small => exp(D) is small
  #=> likelihood at candidate < likelihood at current value
  #The condition log(u) < D will hold for smaller possibilities of u.
  if(log(u) < D){ #u < exp(D) [=ratio of likelihoods] P(log(u)<D)=P(u<exp(D))=exp(D)
    x <- xc
    llhood <- llhoodc
    flag <- flag + 1 #To count number of accepts
  }
  xsam[itr, ] <- x #storing the samples
  #For MALA sampler + univariate case, usually 0.60 is an optimal acceptance rate.
  if(itr %% 100){ #Auto-tuning step
    ar <- flag / itr #Acceptance rate
    if(ar < 0.50){ #Acceptance is too low
```

```
      epsilon <- epsilon / 2
    }
    if(ar > 0.75){ #Acceptance is too high
      epsilon <- epsilon * 2
    }
    print(ar) #To track the acceptance rate
  }
}

xout <- xsam
#Post-burn samples
xp <- xout[(burn+1):Total_itr, ]

#Mean of post-burn samples
colMeans(xp)

#Covariance matrix from samples
cov(xp)

#True covariance matrix
(nu/(nu-2))*Sigma

acf(xp[, 1])
```

After thinning the covariance estimates might improve. Since it is not RW-MH, we required to compute the transition probability (the q1 and q2).

1 Concluding remarks

Here is the list of things to consider:

- Proportional density: MH only requires to know the density in proportional form. This means that your algorithm would not change if you only know the density as $f(\mathbf{x}) \propto \left[1 + \frac{1}{\nu}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right]^{-(\nu+p)/2}$.
- The support of the parameter: Depending on the support of the parameter, you need to decide whether you need a transformed parameter for the MH move. Note that it is better to have the parameter to be unrestricted. If the parameter is already unrestricted, you are good. Otherwise, you may revisit the above table to come up with an appropriate transformation.
- If a parameter is conditionally conjugate, it may be better to use Gibbs sampling than MH.