Capture-Recapture Example (Tutorial)

Leah South

Our running example is about the capture and recapture of the bird species called the European Dipper (*Cinclus cinclus*). Marzolin (1988) collected the data based on the capture and recapture of this species over six years.

Below are the R packages we'll be using in this document.

```
library(MASS) # multivariate normal
library(coda) # assessing convergence and sample quality
library(psych) # bivariate plots
```

The MCMC Algorithm

We'll be looking at MALA (asymptotically un-baised) and ULA (biased) approaches for inference on our example. The focus is on post-processing and bias rather than implementation of algorithms. You can find the relevant code for MALA and ULA implementations for reference.

```
source('MCMC_fn.R')
source('ULA_fn.R')
```

The Statistical Model

The parameters for the model are ϕ_i and p_k where $i = 1, \ldots, 6$ and $k = 2, \ldots, 7$. ϕ_i represents the probability of survival from year i to year $i+1$ and p_k represents the probability of being captured in year k.

The likelihood for the model is given below, and based on data D_i for the number of birds released in year i and y_{ik} for the number of animals caught in year k out of the number released in year i. Here $d_i = D_i - \sum_{k=i+1}^{7} y_{ik}$ is the number released in year i that are never caught. The corresponding probability of a bird being released in year i and never being caught is $\chi_i = 1 - \sum_{k=i+1}^{7} \phi_i p_k \prod_{m=i+1}^{k-1} \phi_m(1-p_m)$, which is a function of the model parameters. The likelihood is given by

$$f(y|\theta) \propto \prod_{i=1}^{6} \chi_i^{d_i} \prod_{k=i+1}^{7} \left[\phi_i p_k \prod_{m=i+1}^{k-1} \phi_m(1-p_m) \right]^{y_{ik}},$$

where $\theta = (\phi, p)$, $\phi = (\phi_1, ..., \phi_6)$, $p = (p_2, ..., p_7)$ and $y = \{y_{ik} : i = 1, \ldots, 6, k = 2, \ldots, 7\}$. Due to parameter identifiability issues, the parameters ϕ_6 and p_7 are combined as $\phi_6 p_7$ leading to a total of eleven parameters.

The prior for each component of θ is set to be $\mathcal{U}(0, 1)$, and all components are independent *a priori*. For the RW proposal, the j-th parameter $\theta[j]$ is transformed using $\tilde{\theta}[j] = \log(\theta[j]/(1 - \theta[j]))$ for $j = 1, \ldots, 11$. The implied prior density for $\tilde{\theta}[j]$ is then $e^{\tilde{\theta}[j]}/(1 + e^{\tilde{\theta}[j]})^2$, for $j = 1, \ldots, 11$.

Read in the liklihood functions and tuning parameters for algorithms

```
load("recapture_ULA_bettertuning.RData")
load("recapture_MALA_tuning.RData")

# Names of the 11 variables
varNames <- paste0("theta",1:11) #### FIX-LATER
```

Multiple Chains

Getting the samples

Let's run 10 chains with a common starting point

```
initial <- c(0.35, -0.66, -1.74, 2.5, -0.67, -0.59, 2.38, 2.52, 1.2, 5.08,
1.3)
set.seed(2)
n_reps <- 10 # number of chains
its <- 500 # number of MCMC iterations
chains_ULA <- chains_MALA <- samples_ULA <- samples_MALA <- list()
for (i in 1:n_reps){

  # Running MALA
  single_chain_mala <- MALA_fn(d = 11, initial = initial, covmala = cov_rw, h = h_mala,
                               iters = its,der_loglike = der_loglike,
                               der_logprior = der_logprior,
                               options = options, varNames = varNames)
  chains_MALA[[i]] <- single_chain_mala
  samples_MALA[[i]] <- single_chain_mala$samples

  # Running ULA
  single_chain_ula <- ULA_fn(d = 11, initial = initial, cov_ULA = cov_ula, h = h_ula,
                               iters = its,der_loglike = der_loglike,
                               der_logprior = der_logprior,
                               options = options, varNames = varNames)
  chains_ULA[[i]] <- single_chain_ula
  samples_ULA[[i]] <- single_chain_ula$samples

}
ula_noburnin <- as.mcmc.list(samples_ULA)
mala_noburnin <- as.mcmc.list(samples_MALA)
save(ula_noburnin,mala_noburnin,chains_MALA,chains_ULA, file = "ten_chains.RData") # For future reuse
```

Compare the KSD

Sourcing in the KSD code and load the KSD package

```
source('KSD.R')
library(KSD)
```

```
## Warning: package 'KSD' was built under R version 4.0.5
```

Evaluate the KSD on each of the chains for MALA and ULA.

```
KSD_MALA_gaussian <- KSD_ULA_gaussian <- rep(NaN,n_reps)
KSD_MALA_imq <- KSD_ULA_imq <- rep(NaN,n_reps)
for (i in 1:n_reps){
  ...
}
save(KSD_MALA_gaussian,KSD_ULA_gaussian,KSD_MALA_imq,KSD_ULA_imq, file = "ksd_ten_chains.RData")
boxplot(KSD_MALA_gaussian,KSD_ULA_gaussian)
boxplot(KSD_MALA_imq,KSD_ULA_imq)
```

Estimating expectations with control variates

Now we'll estimate the posterior expectation of our parameters. The parameters are transformed using $\tilde{\theta}[j] = \log(\theta[j]/(1-\theta[j]))$ for $j = 1, \ldots, 11$. To transform back $\tilde{\theta}[j]$ we use $e^{\tilde{\theta}[j]}/(1+e^{\tilde{\theta}[j]})^2$, for $j = 1, \ldots, 11$.

```
library(ZVCV)
Vanilla_MALA <- ZV1_MALA <- CF_MALA <- SECF_MALA <- matrix(NaN,nrow=n_reps,ncol=d)
Vanilla_ULA <- ZV1_ULA <- CF_ULA <- SECF_ULA <- matrix(NaN,nrow=n_reps,ncol=d)
for (i in 1:n_reps){
  samples <- as.matrix(chains_MALA[[i]]$samples)
  gradients <- chains_MALA[[i]]$der_loglike + chains_MALA[[i]]$der_logprior
  integrand <- 1/(1+exp(-samples))
  Vanilla_MALA[i,] <- colMeans(integrand)
  ...
}
load("Recapture_goldstandard.RData")
# Boxplots of the estimates
for (j in 1:11){
  boxplot(Vanilla_MALA[,j],ZV1_MALA[,j],CF_MALA[,j],SECF_MALA[,j],
          Vanilla_ULA[,j],ZV1_ULA[,j],CF_ULA[,j],SECF_ULA[,j])
  abline(h=gold_standard[j])
}
```

Investigate tuning parameter for ULA

Try increasing and decreasing the parameter h_ula and investigating the KSD.

Investigate convergence for MALA

Try alternative initialisation points you may use the following to simulate from the prior:

```
initial <- recapture_simprior(1, options)
```

Investigate the convergence using multiple chains and Gelman & Rubin's $\hat{R}$ diagnostic. Work out a good burnin and investigate control variates on the burnin chain