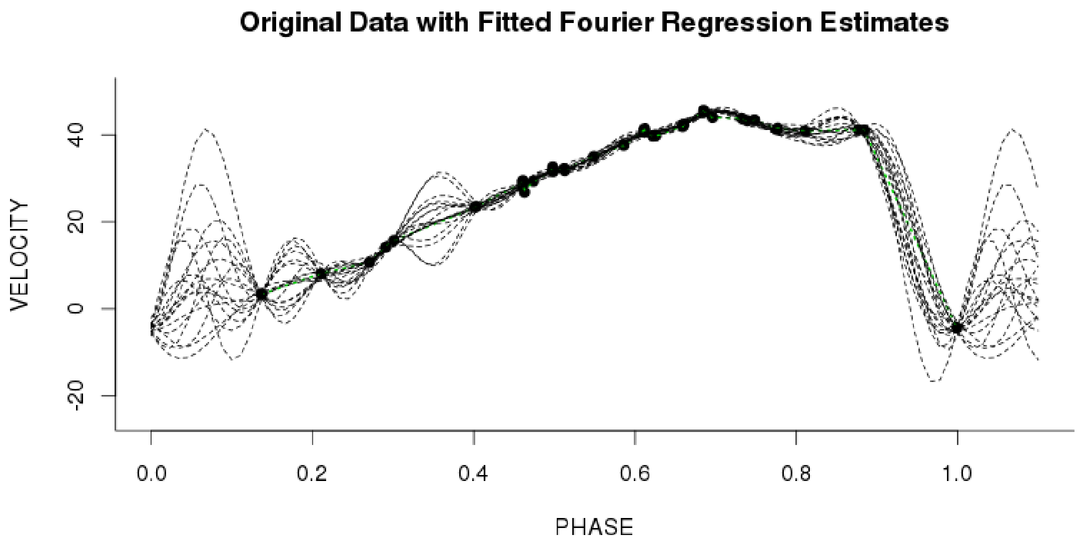
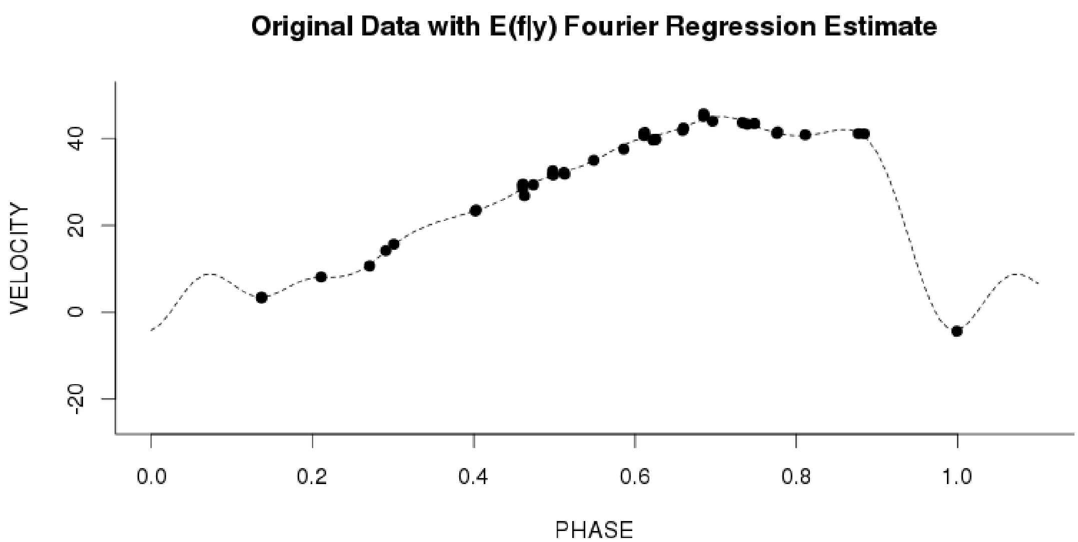
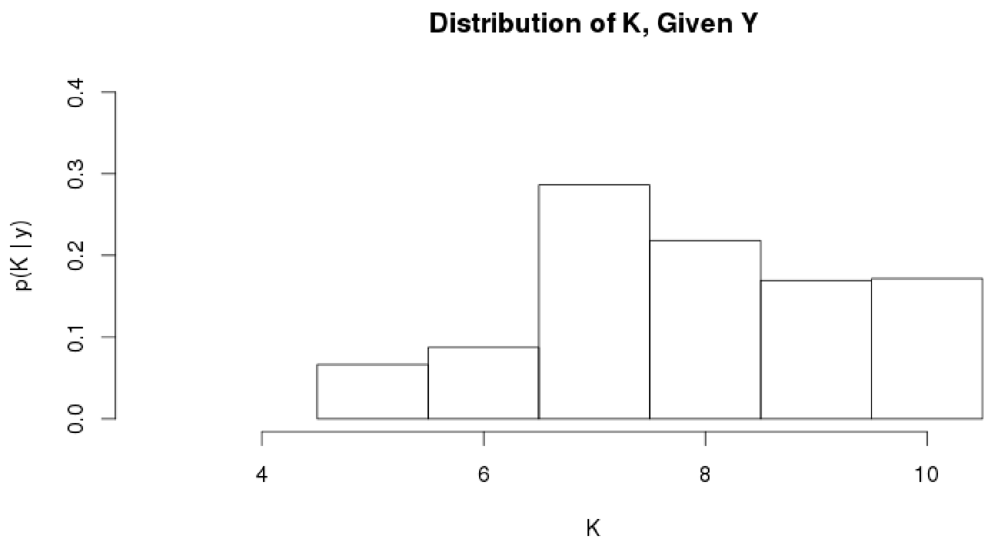
MCMC Midterm 2: Attachments

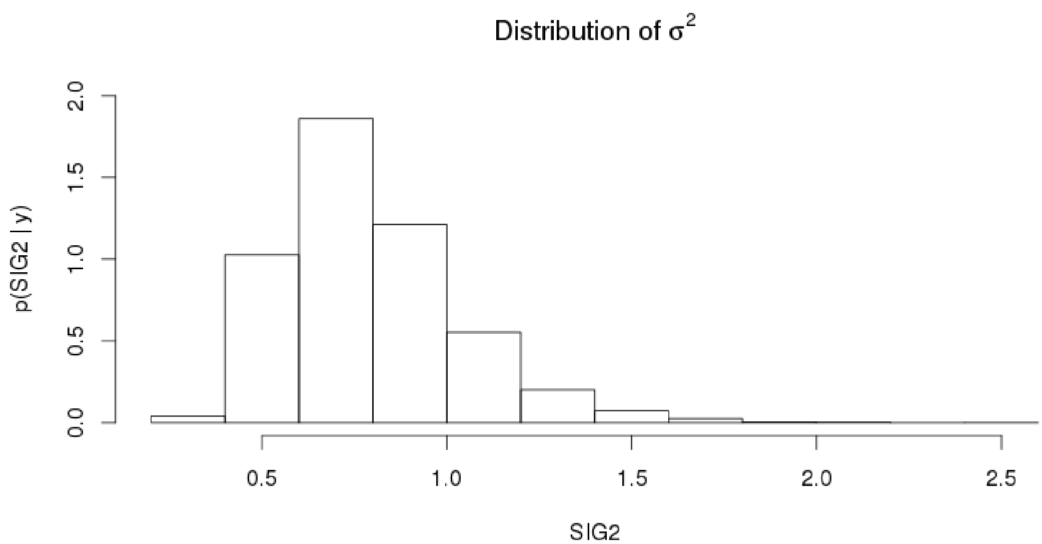
Maurice Diesendruck

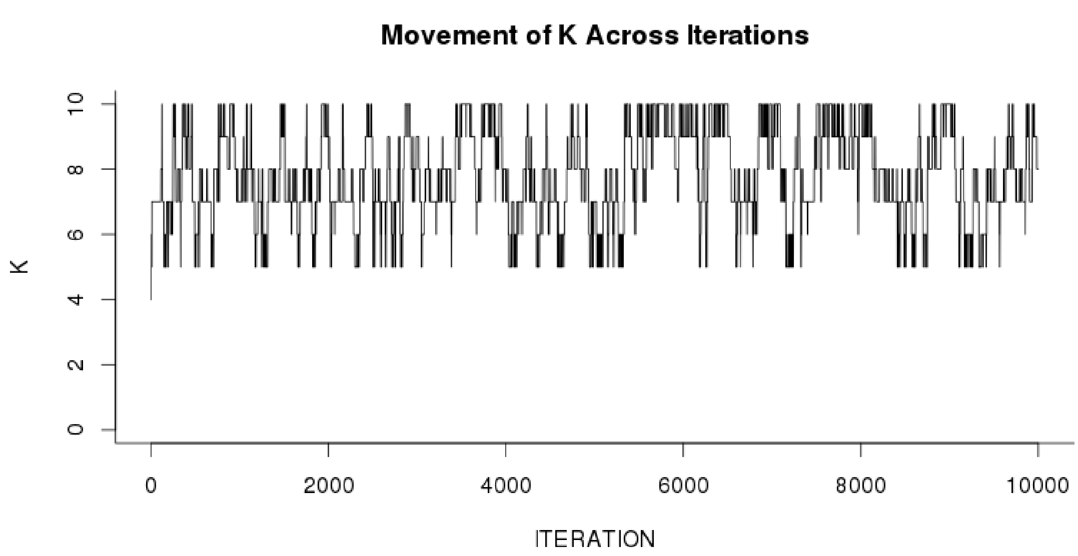
April 10, 2015

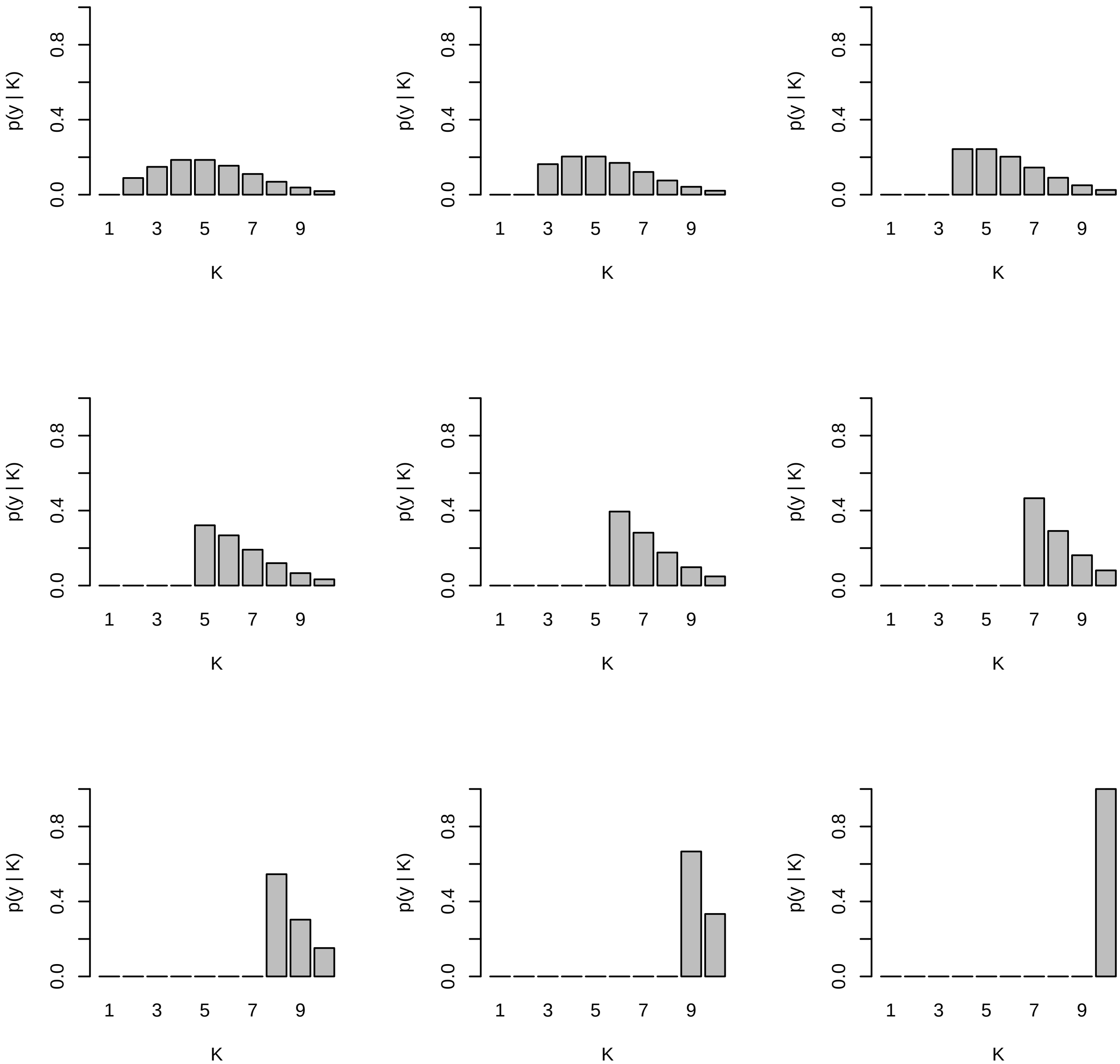


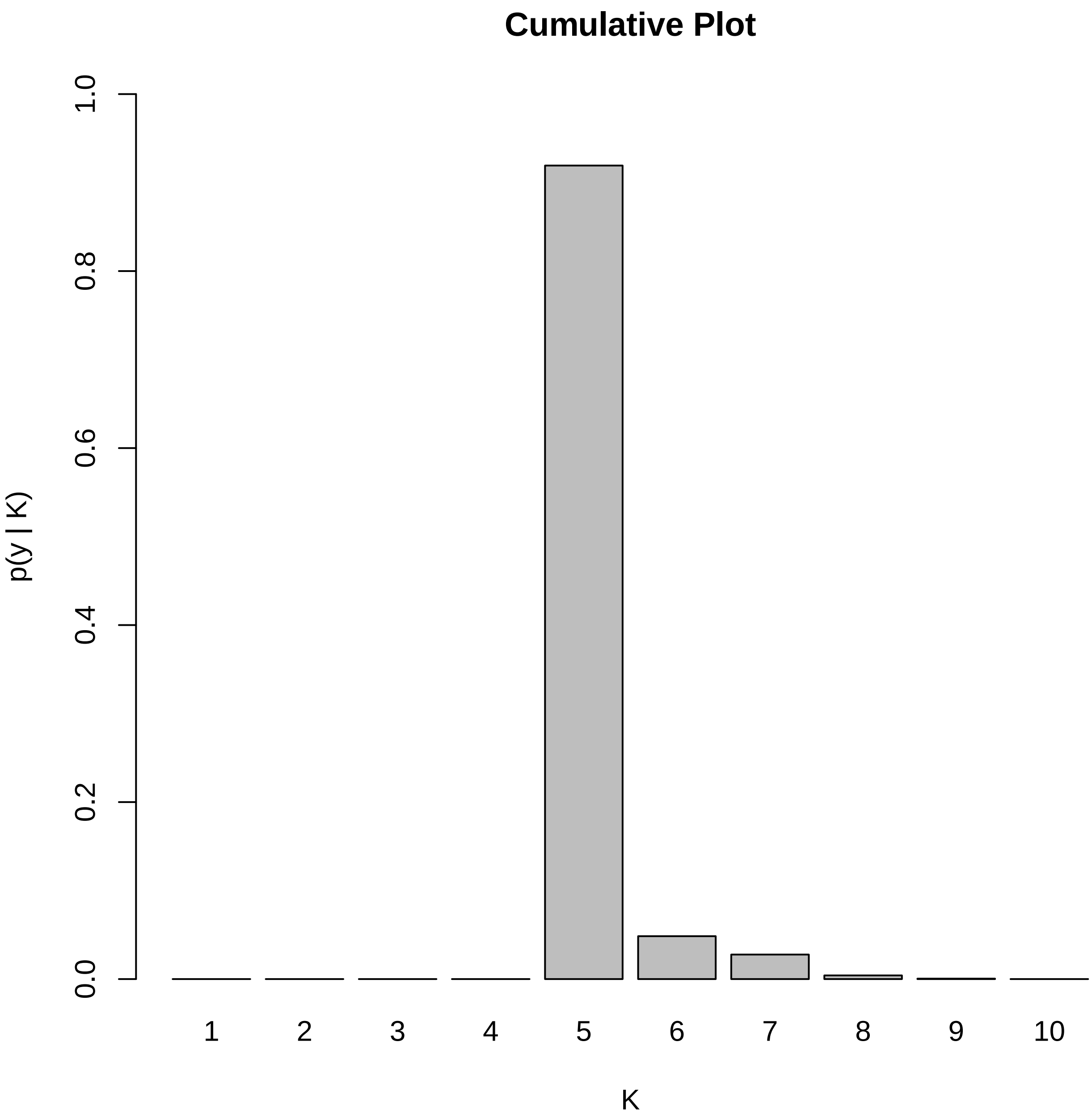



Distribution of K, Given Y
p(K | y)
0.0
0.1
0.2
0.3
0.4
4
6
8
10
K


Distribution of σ2
p(SIG2 | y)
0.0
0.5
1.0
1.5
2.0
0.5
1.0
1.5
2.0
2.5
SIG2


Movement of K Across Iterations
K
0
2
4
6
8
10
0
2000
4000
6000
8000
10000
ITERATION


p(y | K)
0.0
0.4
0.8
1
3
5
7
9
K



Relevant Code Snippets

0.1 Log Joint Posterior for Ratio

```
ljointpost=function(K, b, sig2){
  #print("calculating posterior with param:");
  n = length(y);
  p <- length(b)
  yhat <- X[,1:p] %*% b
```

```
  # Posterior. Cancel out terms without K, yhat, b.
  lpo  <- (-1/2)*(1/sig2)*sum((y-yhat)^2) - 1/2*sum(b^2)/10 + K*log(lambda) -
           log(factorial(K))

  return(lpo);
}
```

0.2 Gibbs Conditional Posterior Distributions

```
sample.b <- function(K,sig2) { # generate b ~ p(b | K, sig2, y)
  idx <- 1:(2*K+1)    # select columns (elements) for K harmonics
  Xk <- X[,idx]       # Subset of design matrix, with 2K+1 columns.
  A0k <- A0[idx,idx] # Precision matrix of beta prior.
  b0k <- b0[idx]      # Mean vector of zeros from beta prior.

  # Full conditionals from Question 2
  V <- solve(t(Xk)%*%Xk/sig2+A0k)
  mm <- V%*%(t(Xk)%*%y/sig2)
  L <- t( chol(V))              # LL' = V
  b <- mm + L %*% rnorm(2*K+1) # b ~ N(m,V)

  return (b)
}

sample.sig2 <- function(K,b) { # generate 1/sig2 ~ p(1/sig2 | K,b,y)
  p <- length(b)
  idx <- 1:(2*K+1)    # select columns (elements) for K harmonics
  Xk <- X[,idx]       # Subset of design matrix, with 2K+1.

  a1 <- (n/2)+1
  b1 <- t(y-Xk%*%b)%*%(y-Xk%*%b)/2 + 1

  sig2inv <- rgamma(1,shape=a1,rate=b1)
  sig2 <- 1/sig2inv
  return (sig2)
}
```

0.3 Auxiliary Variable Transformation and Reverse Transformation

```
qu <- function(K, b, sig2) {
  ## find m,L for mapping T: (b,u) -> b1,  below
  ## bnew = m + L*u,
  ## Use a regression of residuals on (K+1)-st harmonic
```

```
  ##     to determine m and L
  idx <- 1:(2*K+1)   # select columns (elements) for K harmonics
  Xk <- X[,idx]      # Subset of design matrix, with columns for setting of K.
  eps <- y-Xk%*%b

  regression <- lm(eps ~ sin((K+1)*2*pi*x) + cos((K+1)*2*pi*x))
  mk <- regression$coefficients[2:3]
  Vk <- vcov(regression)[2:3,2:3]
  Lk <- t(chol(Vk))
  return (list(m=mk, V=Vk, L=Lk))
}

Tinv <- function(K1,b1,sig2) {
  ## proposed (shorter) par vector
  ## bnew = m + Lu or u = L^-1 (bnew-m)
  K <- K1-1
  p <- 2*K+1
  b <- b1[1:p]
  bnew <- b1[c(p+1,p+2)]

  ## back out auxiliary u, and logJ
  fit <- qu(K, b, sig2)
  u <- solve(fit$L)%*%(bnew-fit$m)
  logJ <- sum(log(diag(fit$L)))

  return (list(b=b, u=u, logJ=logJ))
}
```

0.4 Acceptance Probability for Birth Move

```
rho <- function(K, b1, b, u, logJ, sig2) {
  ##  acceptance ratio for birth move,
  ##  moving from b -> (b,u)
  # current parameter: b
  # proposal:          b1
  K1 <- K+1

  lqu <- sum(dnorm(u, m=0, sd=1, log=TRUE)) # TODO: Shouldn't this be dnorm?

  ljointpostratio <- ljointpost(K1, b1, sig2) - ljointpost(K, b, sig2)

  # Priors and likelihood
  rho <- exp(ljointpostratio)

  # Transition probabilities
```

```
  if (K==1) {
    rho <- rho*2
  } else if (K==Kmx-1) {
    rho <- rho/2
  } else {
    # Scaling rule smoothely penalizes larger values of K.
    birth.weight <- qbeta((Kmx-K+0.01)/Kmx, 1, 5)
    rho <- rho*birth.weight
  }
  rho <- rho/exp(lqu)          # Auxiliary
  rho <- rho*exp(logJ)         # Jacobian

  return (rho)
}
```