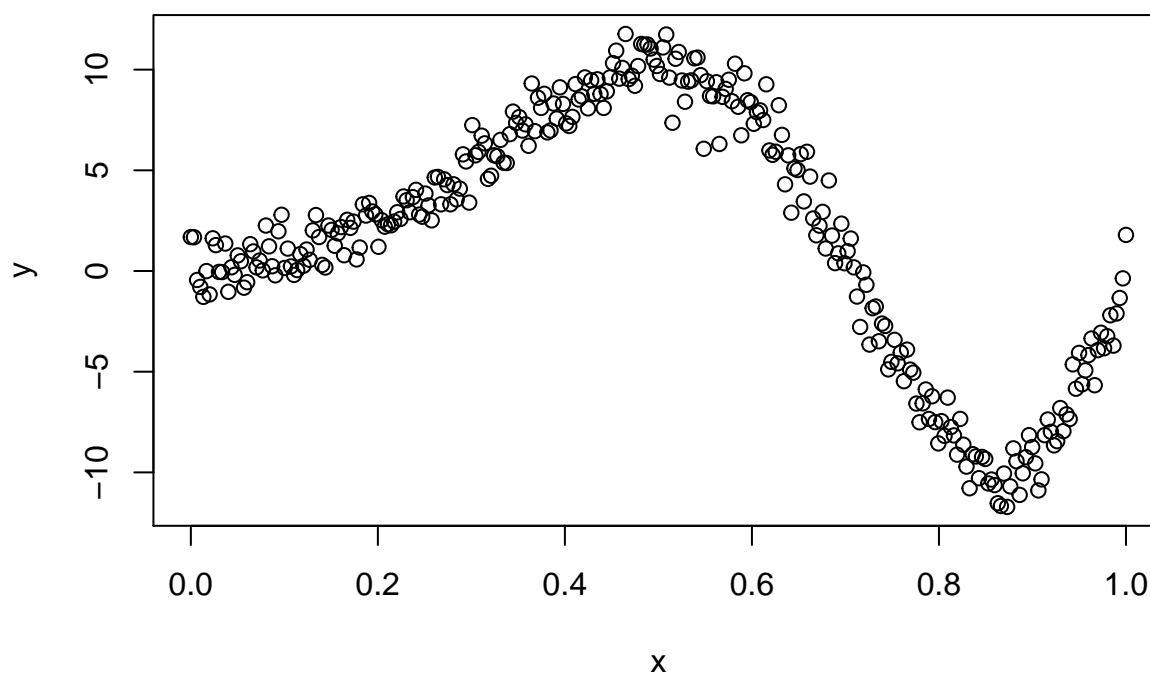


Bayesian Regression Splines

Andy Shen, Devin Francom

Let's say you want to fit a model using some wiggly data. Maybe

```
n<-300
x<-seq(0,1,length.out=n)
y<-sin(2*pi*x^2)*10+rmnorm(n)
plot(x,y)
```



One way to fit a model to data like this is to come up with a linear basis and fit a linear model using the basis as the X matrix (which we will call B). People often use splines as a basis. The simplest set of spline basis functions would be to make the i th basis function (i.e., the i th column of B) look like

$$B_{ij} = [s_i(x_j - t_i)]_+$$

where $s \in \{-1, 1\}$, which we'll call the sign, and t is a value in the domain of x , which we will call a knot. Also, $[a]_+ = \max(0, a)$.

Try some combinations of s and t to see what your basis functions look like, and what the corresponding linear model fit looks like (using the `lm` function or your Bayesian linear model code). Try with different numbers of basis functions, also.

Bayesian Spline with Random Knots

```
knotnum <- 5
its = 3000
source("mcmc_spline_fixed.R")

res <- mcmc_spline(knotnum)
mat_beta <- res[[1]]
mat_sig <- res[[2]]
mat_t <- res[[3]]
X_curr <- res[[4]]
ar <- res[[5]]
mat_s <- res[[6]]

colMeans(mat_beta)

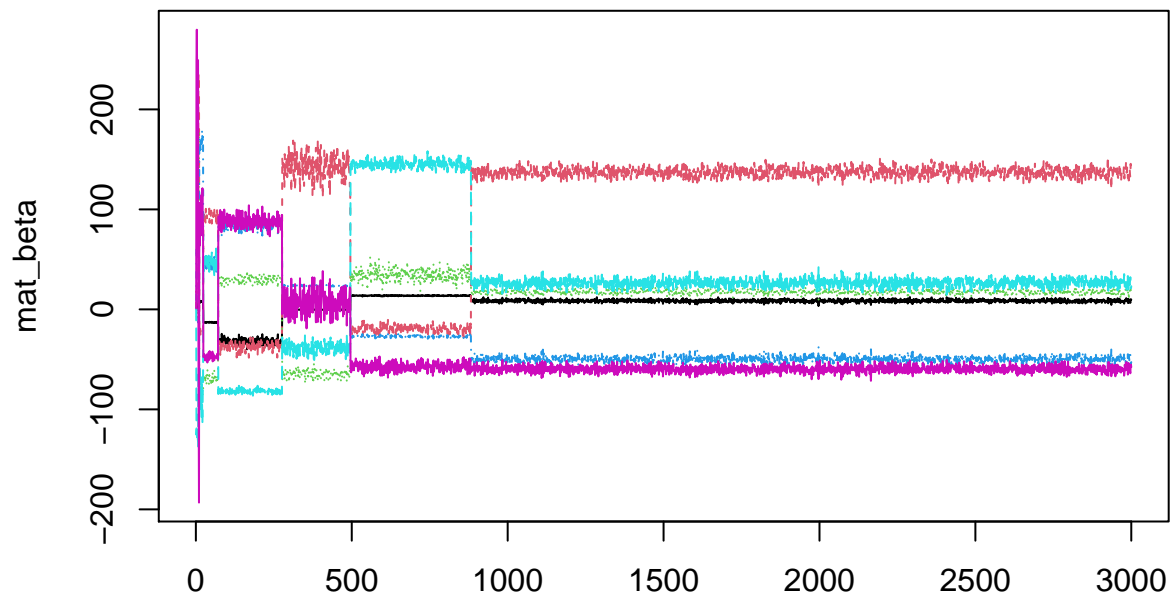
## [1] 5.442692 103.954701 12.813780 -29.783868 28.982183 -42.895519
colMeans(mat_t)

## [1] 0.8011626 0.2518108 0.4927853 0.8004192 0.6218184
mean(mat_sig)

## [1] 1.08888
ar

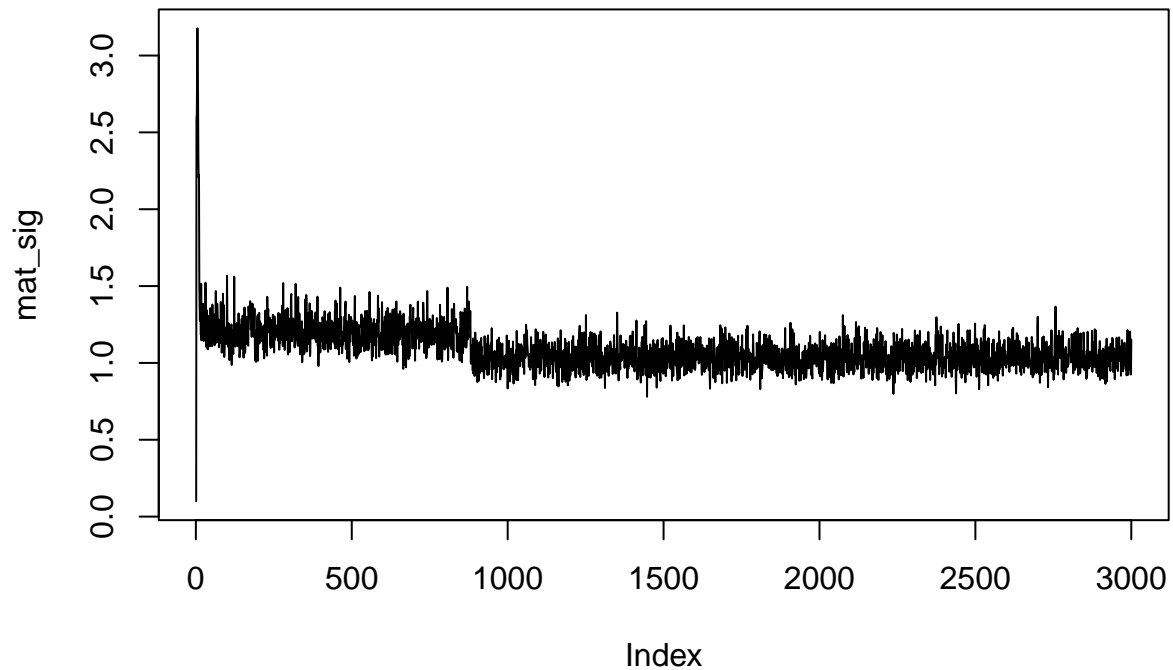
## [1] 0.002666667
matplot(mat_beta, type = "l", main = "Plot of Regression Coefficients")
```

Plot of Regression Coefficients



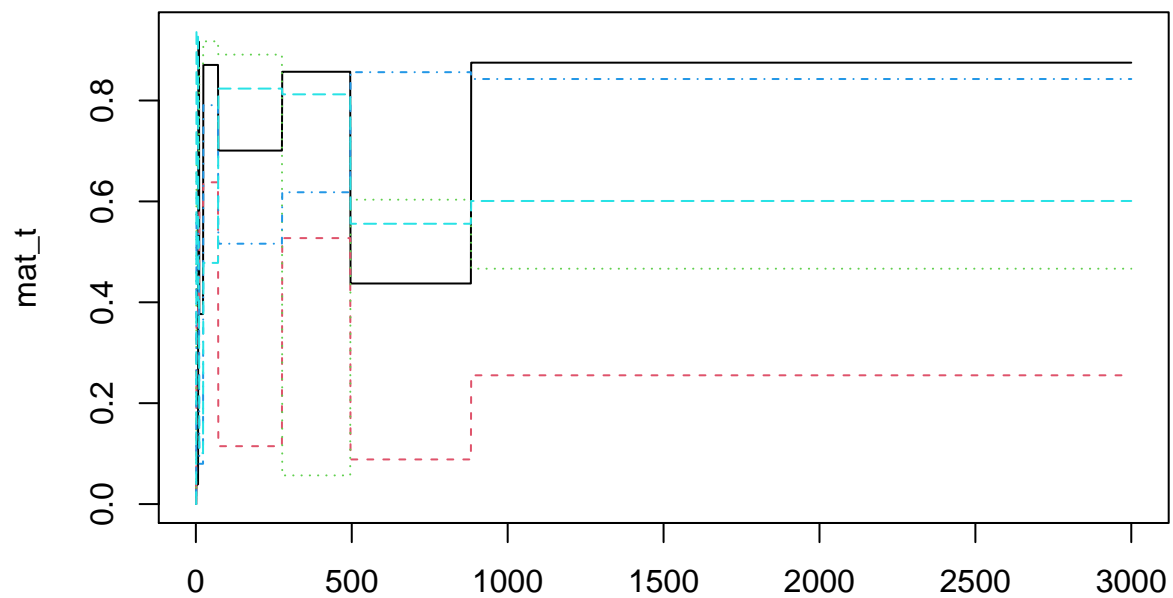
```
plot(mat_sig, type = "l", main = "Plot of Sigma^2")
```

Plot of Sigma^2



```
matplot(mat_t, type = "l", main = "Plot of Knot Locations")
```

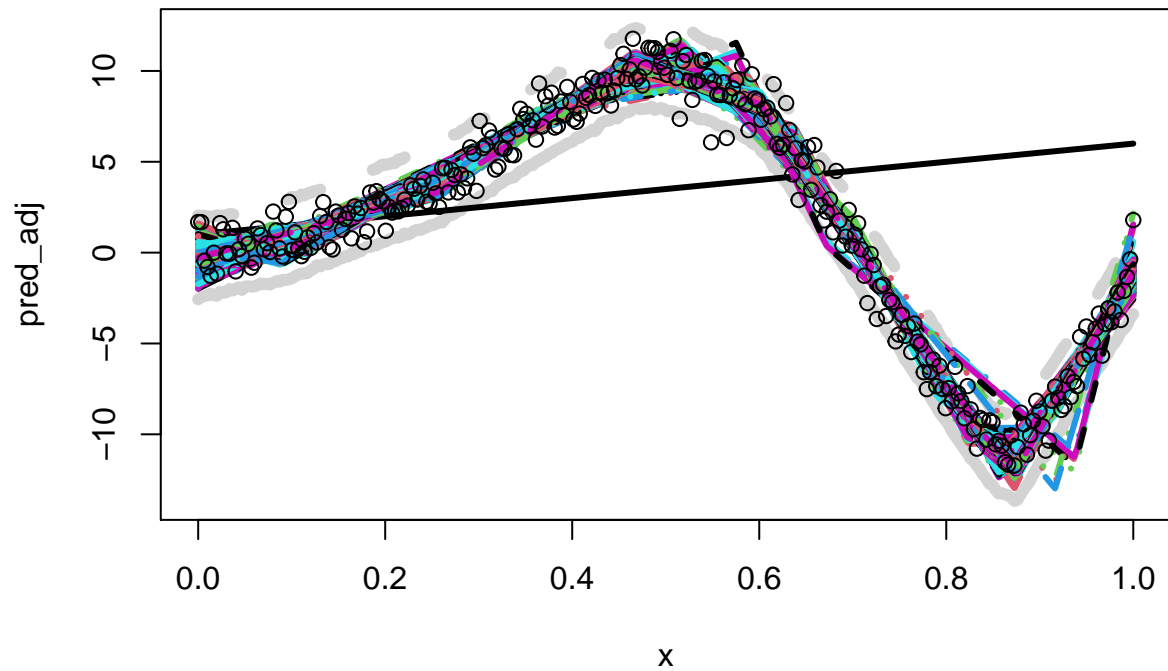
Plot of Knot Locations



```
mean.pred <- matrix(NA, nrow = its, ncol = length(x))
pred <- mean.pred
for(p in 1:its) {
  mean.pred[p,] <- spline.basis(nknot = knotnum, knots = mat_t[p,], signs = mat_s[p,]) %*% mat_beta[p,]
  pred[p,] <- mean.pred[p,] + rnorm(length(x), sd = sqrt(mat_sig[p]))
}
```

```
mean.pred <- t(mean.pred)
```

```
pred_adj <- t(apply(pred, 2, quantile, probs = c(0.025, 0.975)))
matplot(x, pred_adj, col = "lightgrey", lwd = 6, type = "l")
matplot(x, mean.pred, type = "l", lwd = 3, add = TRUE)
points(x, y)
```



```
plot(x, rowMeans(mean.pred), lwd = 10, type = "l", col = "royalblue")
points(x, y)
```

