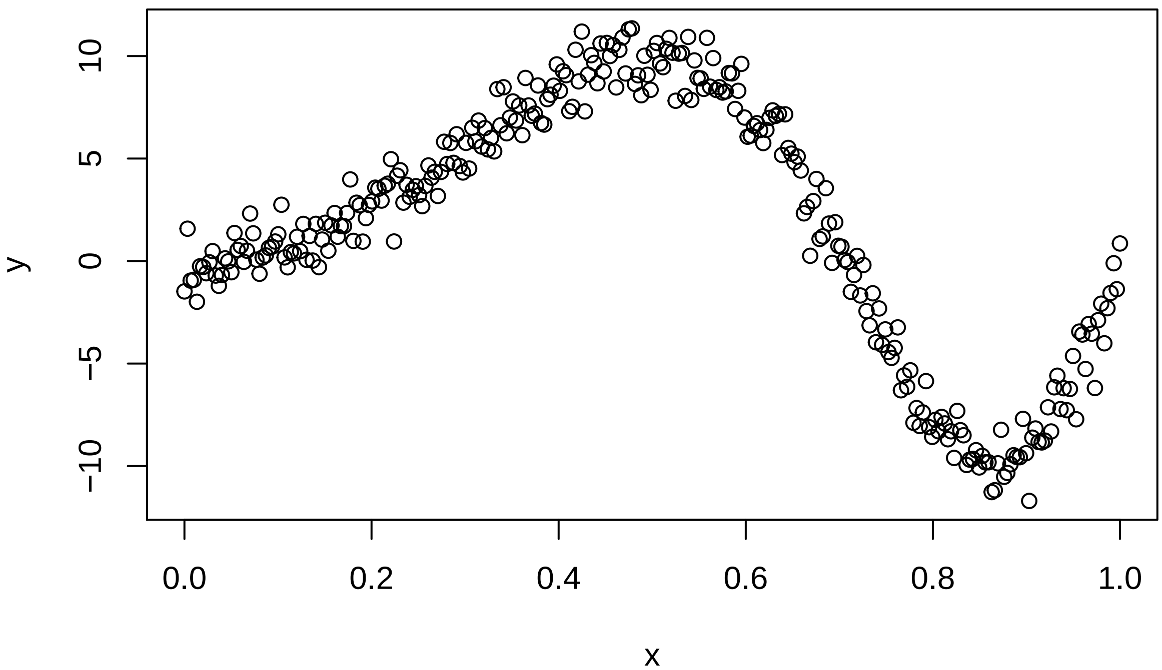
Introduction to Splines

Andy Shen, Devin Francom

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

```r
set.seed(12)
n<-300
x<-seq(0,1,length.out=n)
y<-sin(2*pi*x^2)*10+rnorm(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 ith basis function (i.e., the ith column of B) look like

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

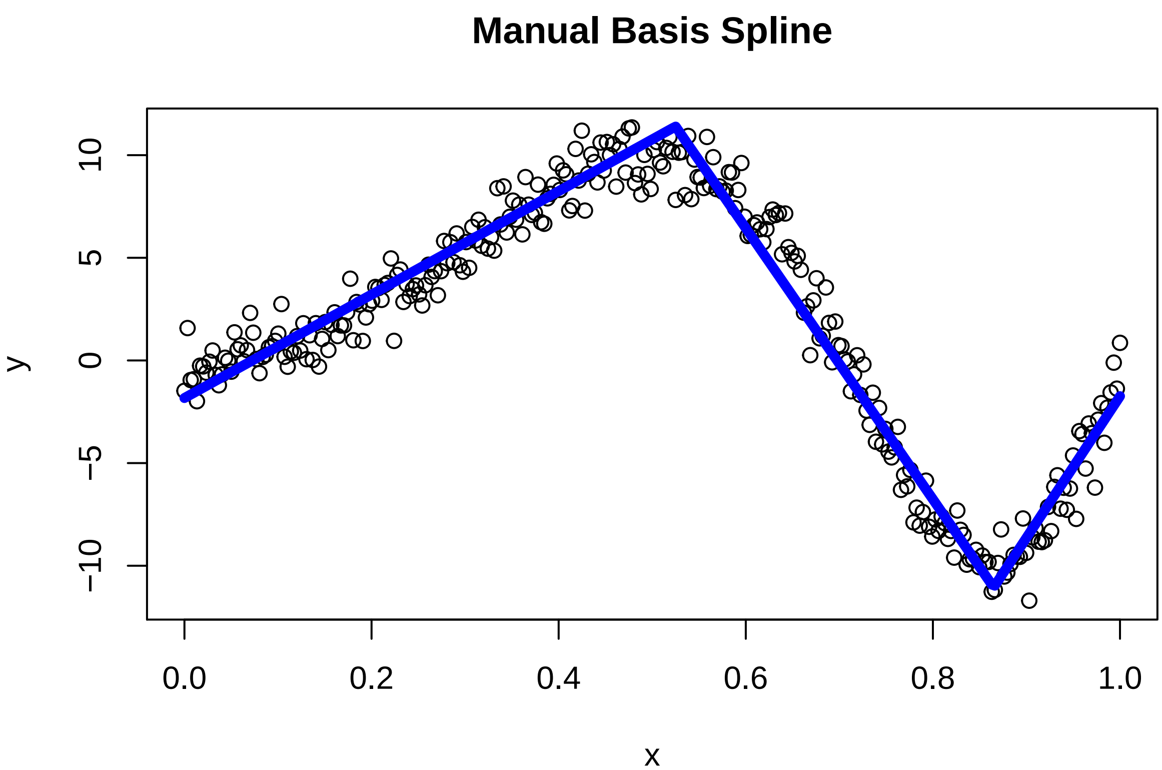
where $s \in \{-1, 1\}$, which we'll call the sign, and t is a value in the domain of x, which we will call a knot. Also, $[a]_+ = max(0, a)$.

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.

1 Manual Frequentist Spline Function

```
generate_spline <- function(tvec, y, x, nknot = length(tvec)) {
  s <- sample(c(1), nknot, replace = TRUE)
  Bmat <- matrix(NA, nknot, length(x))
  hs <- Bmat

  for(i in 1:nknot) {
    for(j in 1:length(x)) {
      Bmat[i,j] <- max(s[i] * (x[j] - tvec[i]), 0)
    } #creating basis functions
  }

  mBmat <- t(Bmat)
  mod <- lm(y ~ mBmat) #use gibbs to sample coefs in bayes
  pred <- predict(mod)

  sq <- x
  for(ii in 1:nknot) {
    if(s[i] == 1) {
      hs[ii,] <- sq - tvec[ii]
      hs[ii,][sq < tvec[ii]] <- 0
    }
    else {
      hs[ii,] <- -1*(sq - tvec[ii])
      hs[ii,][sq < tvec[ii]] <- 0
    }
  } #setting x values

  plot(x,y, main = "Manual Basis Spline")
  lines(x, pred, type = "l", lwd = 5, col="blue1")
  summary(mod)
}
```

```
tv <- c(0, 0.525, 0.865) #vector of t-values
generate_spline(tv, y = y, x = x)
```



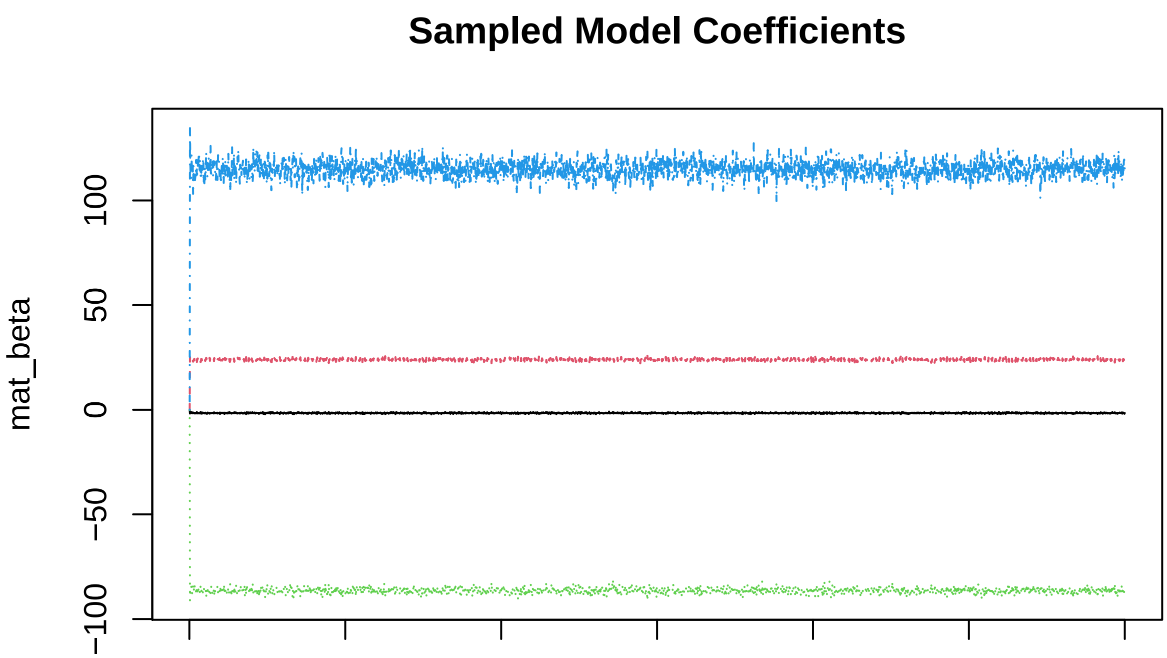

```
##
## Call:
## lm(formula = y ~ mBmat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5715 -0.8087 -0.0387  0.7499  3.4953
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.8306     0.1817  -10.08   <2e-16 ***
## mBmat1       25.1946     0.5270   47.81   <2e-16 ***
## mBmat2      -91.2259     1.1975  -76.18   <2e-16 ***
## mBmat3      135.0260     3.6954   36.54   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.208 on 296 degrees of freedom
## Multiple R-squared:  0.963,  Adjusted R-squared:  0.9626
## F-statistic:  2566 on 3 and 296 DF,  p-value: < 2.2e-16
```

2 Bayesian Spline

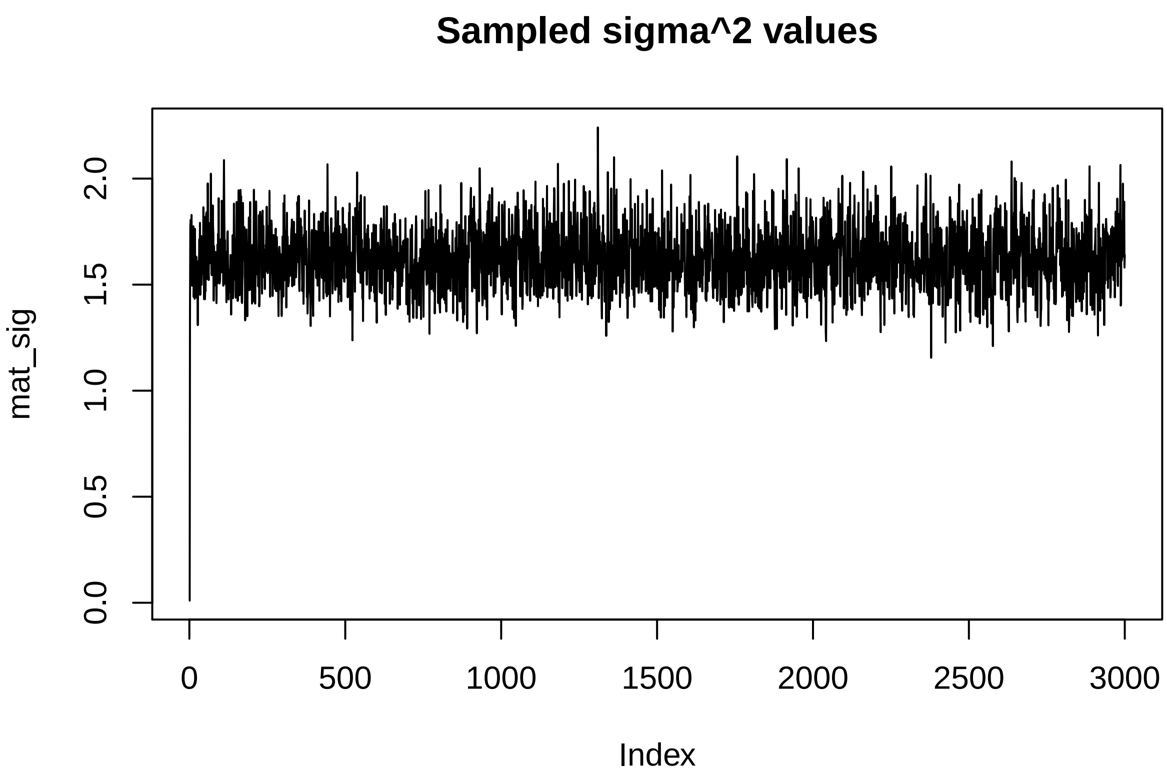
```r
library(mvtnorm)
bayes_spline <- function(tvec, y, x, nknot = length(tvec)) {
  s <- sample(c(1), nknot, replace = TRUE)
  Bmat <- matrix(NA, nknot, length(x))
  hs <- Bmat

  for(i in 1:nknot) {
    for(j in 1:length(x)) {
      Bmat[i,j] <- max(s[i] * (x[j] - tvec[i]), 0)
    } #creating basis
  } #X Matrix

  X <- t(Bmat)
  X <- cbind(1, X)

  p_sig <- function(a = 0, b = 0, n = nrow(X), beta) {
    a_term <- a + (n/2)
    b_term <- 0.5 * (2*b + (t(y - (X %*% beta)) %*% (y - (X %*% beta)))) #y, X defined above
    1 / rgamma(1, shape = a_term, rate = b_term)
  }

  p_beta <- function(sig_sq, tau_sq = 100, p = ncol(X)-1) {
    sig <- solve( (1/sig_sq) * (t(X) %*% X) + (1/tau_sq) * diag(p+1) )
    mu <- (1/sig_sq) * sig %*% t(X) %*% y
    rmvnorm(1, mean = mu, sigma = sig)
  }


  gibbs <- function(its) {
    mat_beta <- matrix(NA, its, ncol(X))
    mat_sig <- rep(NA, its)
    mat_sig[1] <- 0.01
    mat_beta[1,] <- rep(0, ncol(X))
    for(it in 2:its) {
      mat_beta[it,] <- p_beta(sig_sq = mat_sig[it-1])
      mat_sig[it] <- p_sig(beta = mat_beta[it,])
    }
    list(mat_beta, mat_sig)
  }

  its <- 3000
  a <- gibbs(its = its)
  mat_beta <- a[[1]] #beta values
  mat_sig <- a[[2]] #sig^2 values

  sq <- x
  for(ii in 1:nknot) {
    if(s[i] == 1) {
      hs[ii,] <- sq - tvec[ii]
      hs[ii,][sq < tvec[ii]] <- 0
    }
```

```
    else {
      hs[ii,] <- -1*(sq - tvec[ii])
      hs[ii,][sq > tvec[ii]] <- 0
    }
  } #setting x values

  list(mat_beta, mat_sig)
}
mat_beta <- bayes_spline(c(0, 0.525, 0.865), y = y, x = x)[[1]]
mat_sig <- bayes_spline(c(0, 0.525, 0.865), y = y, x = x)[[2]]
```

```
matplot(mat_beta, type = "l", main = "Sampled Model Coefficients")
```




```
plot(mat_sig, type = "l", main = "Sampled sigma^2 values")
```




```r
burn <- 1:15
colMeans(mat_beta[-burn,]) #target: -1.83, 25.19, -91.23, 135.03
```

```
## [1]  -1.576187  23.966745 -86.398181 114.966455
```

```r
mean(mat_sig[-burn]) #target: 1.46
```

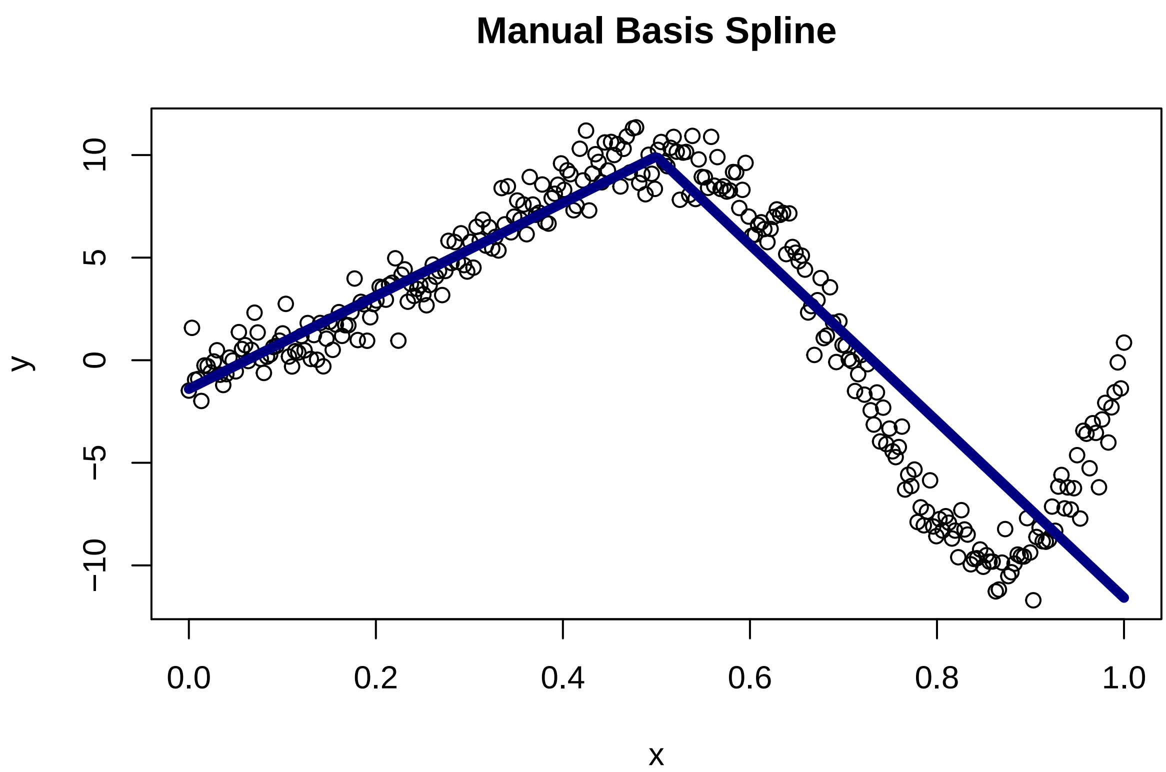
```
## [1] 1.62259
```

Trying things out

```
t1 <- 0.5 #knot at 0.5
s <- 1
B1 <- rep(NA, length(x))

for(i in 1:length(x)) {
  B1[i] <- max(s * (x[i] - t1), 0)
}
mod <- lm(y ~ x + B1)
summary(mod)
```

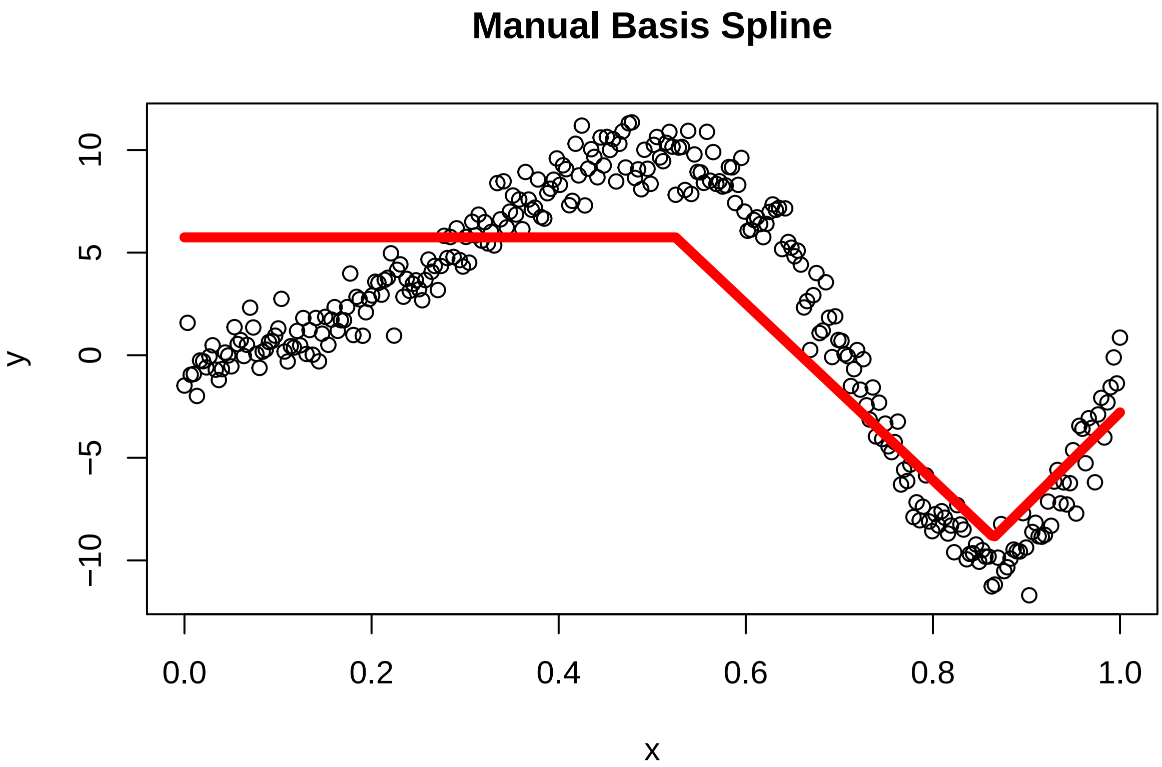
```
##
## Call:
## lm(formula = y ~ x + B1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.8060 -1.0370  0.0118  1.3000 12.4394
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.4034     0.4206  -3.336 0.000957 ***
## x            22.6785     1.2327  18.398  < 2e-16 ***
## B1          -65.7131     2.2051 -29.801  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.766 on 297 degrees of freedom
## Multiple R-squared:  0.8054, Adjusted R-squared:  0.8041
## F-statistic: 614.5 on 2 and 297 DF,  p-value: < 2.2e-16
```

```
cf <- mod$coefficients
sq <- x
hs <- sq - t1
hs[sq < t1] <- 0
yfit <- cf[1] + cf[2]*x + cf[3]*hs

plot(x,y, main = "Manual Basis Spline")
lines(x, yfit, type = "l", lwd = 5, col="navy")
```



Add another knot

```
t1 <- 0.525
t2 <- 0.865
s <- 1
B1 <- rep(NA, length(x))
B2 <- B1

for(i in 1:length(x)) {
  B1[i] <- max(s * (x[i] - t1), 0)
  B2[i] <- max(s * (x[i] - t2), 0)
}
mod <- lm(y ~ B1 + B2)

cf <- mod$coefficients
sq <- x
hs1 <- sq - t1
hs1[sq < t1] <- 0

hs2 <- sq - t2
hs2[sq < t2] <- 0

yfit <- cf[1] + cf[2]*x + cf[3]*hs1 + cf[4]*hs2
yfit2 <- predict(mod) #same thing

plot(x,y, main = "Manual Basis Spline")
lines(x, yfit2, type = "l", lwd = 5, col="red")
```



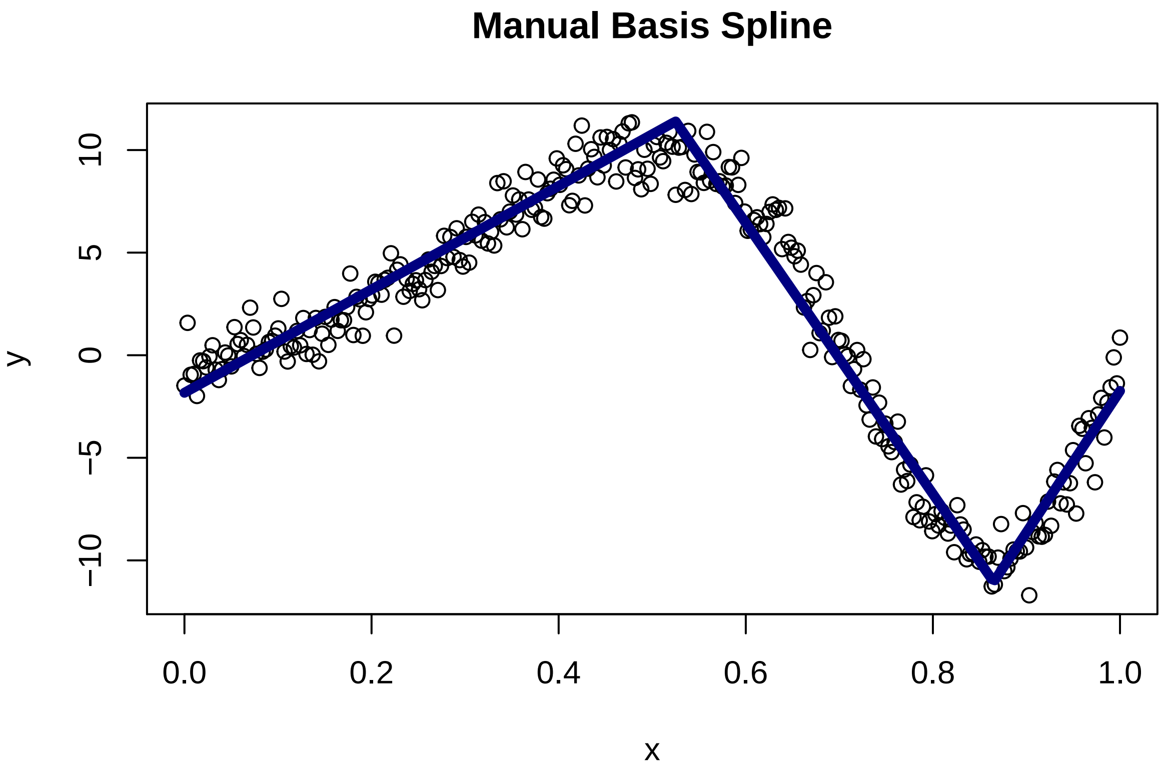
Manual Basis Spline
y
10
5
0
−5
−10
0.0
0.2
0.4
0.6
0.8
1.0
x

Add another knot (expected)

```r
t1 <- 0 #knot at 0.5
t2 <- 0.525 #another knot at 0.85
t3 <- 0.865
s <- 1
B1 <- rep(NA, length(x))
B3 <- B2 <- B1

for(i in 1:length(x)) {
  B1[i] <- max(s * (x[i] - t1), 0)
  B2[i] <- max(s * (x[i] - t2), 0)
  B3[i] <- max(s * (x[i] - t3), 0)
}
mod <- lm(y ~ B1 + B2 + B3)
summary(mod)
```

```
##
## Call:
## lm(formula = y ~ B1 + B2 + B3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5715 -0.8087 -0.0387  0.7499  3.4953
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.8306     0.1817  -10.08   <2e-16 ***
## B1           25.1946     0.5270   47.81   <2e-16 ***
## B2          -91.2259     1.1975  -76.18   <2e-16 ***
## B3          135.0260     3.6954   36.54   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.208 on 296 degrees of freedom
## Multiple R-squared:  0.963,  Adjusted R-squared:  0.9626
## F-statistic:  2566 on 3 and 296 DF,  p-value: < 2.2e-16
```

```r
cf <- mod$coefficients
sq <- x

hs1 <- sq - t1
hs1[sq < t1] <- 0

hs2 <- sq - t2
hs2[sq < t2] <- 0

hs3 <- sq - t3
hs3[sq < t3] <- 0

yfit2 <- cf[1] + cf[2]*x + cf[3]*hs1 + cf[4]*hs2 + cf[5]*hs3
yfit <- predict(mod)
plot(x,y, main = "Manual Basis Spline")
lines(x, yfit, type = "l", lwd = 5, col="navy")
```

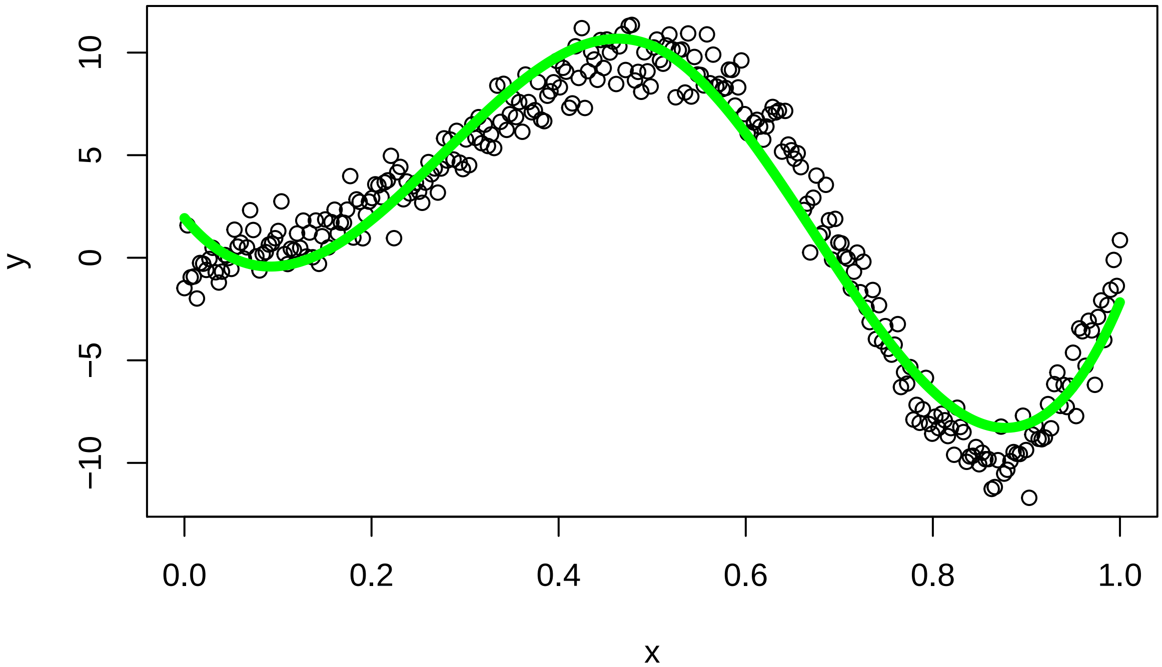


Manual Basis Spline
y
10
5
0
−5
−10
0.0
0.2
0.4
0.6
0.8
1.0
x

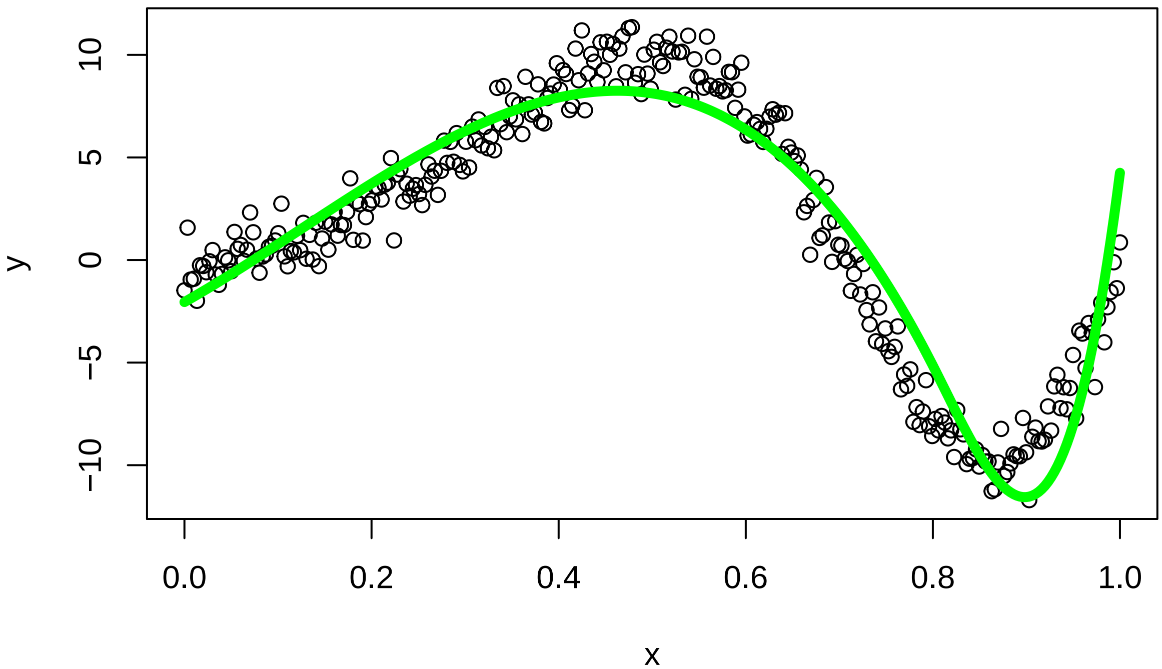
Using the `bs()` Function

1 Knot

```
library(splines)
df <- data.frame(y, x)
m2 <- lm(y ~ bs(x, knots = 0.5), data = df)
pred <- predict(m2)

plot(x,y)
lines(x, pred, lwd = 5, col = "green")
```

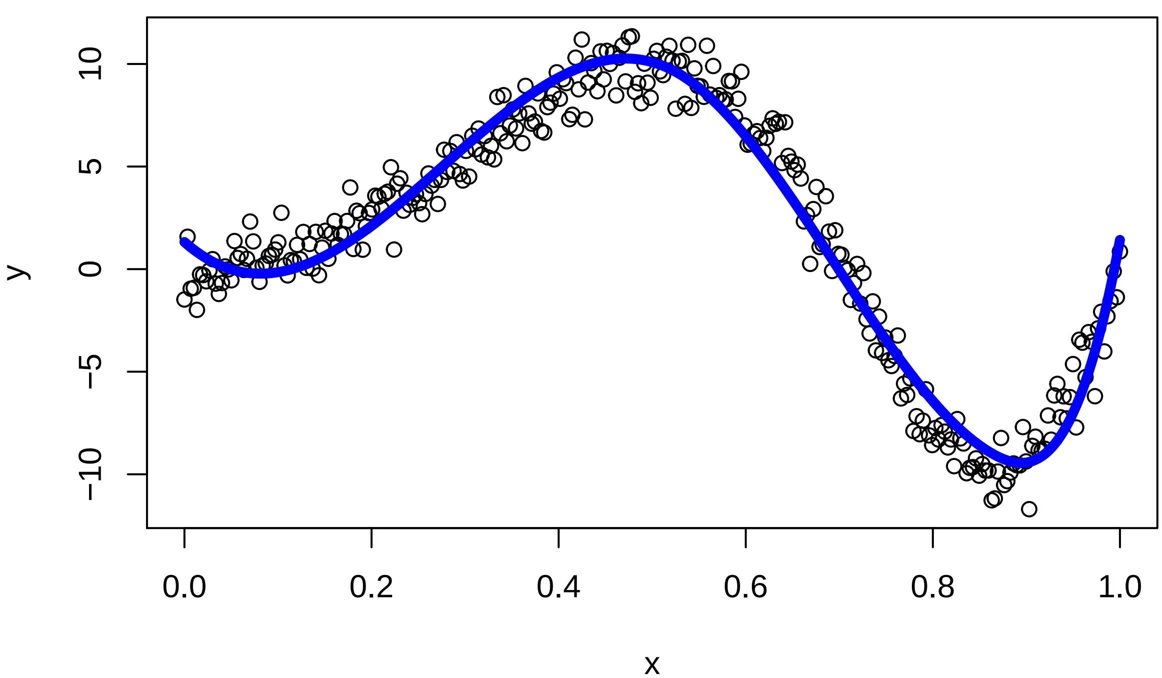



```
m2 <- lm(y ~ bs(x, knots = 0.8), data = df)
pred <- predict(m2)

plot(x,y)
lines(x, pred, lwd = 5, col = "green")
```



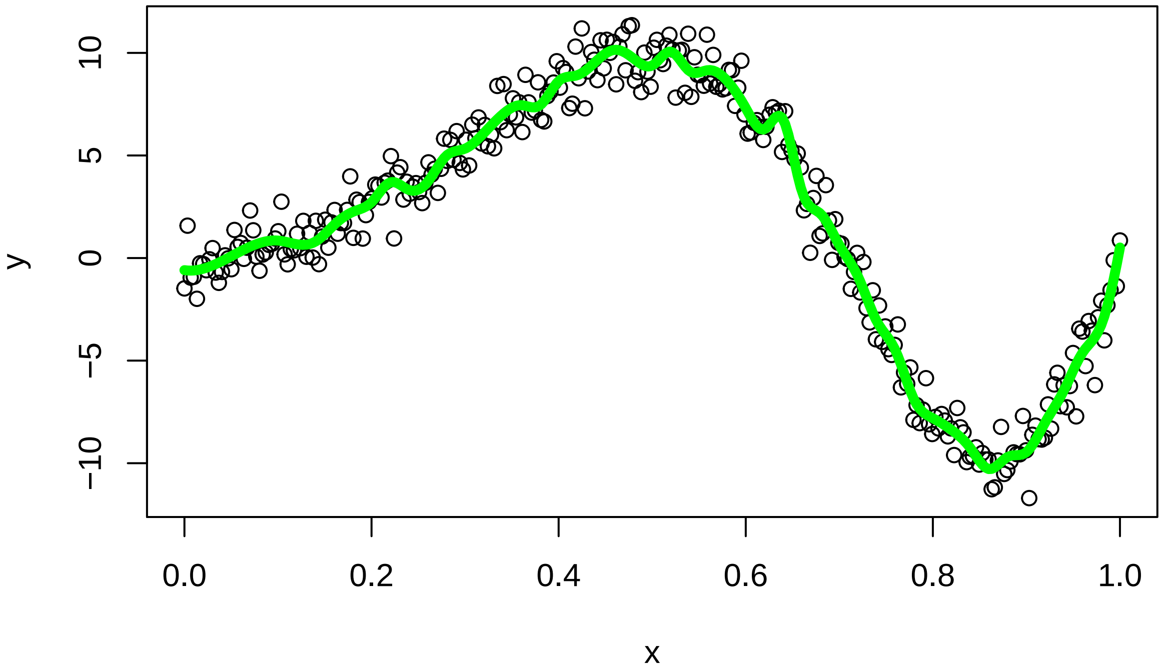
2 Knots (Expected)

```
m1 <- lm(y ~ bs(x, knots = c(0.525, 0.865)), data = df)
pred <- predict(m1)

plot(x,y)
lines(x, pred, lwd = 5, col = "blue")
```



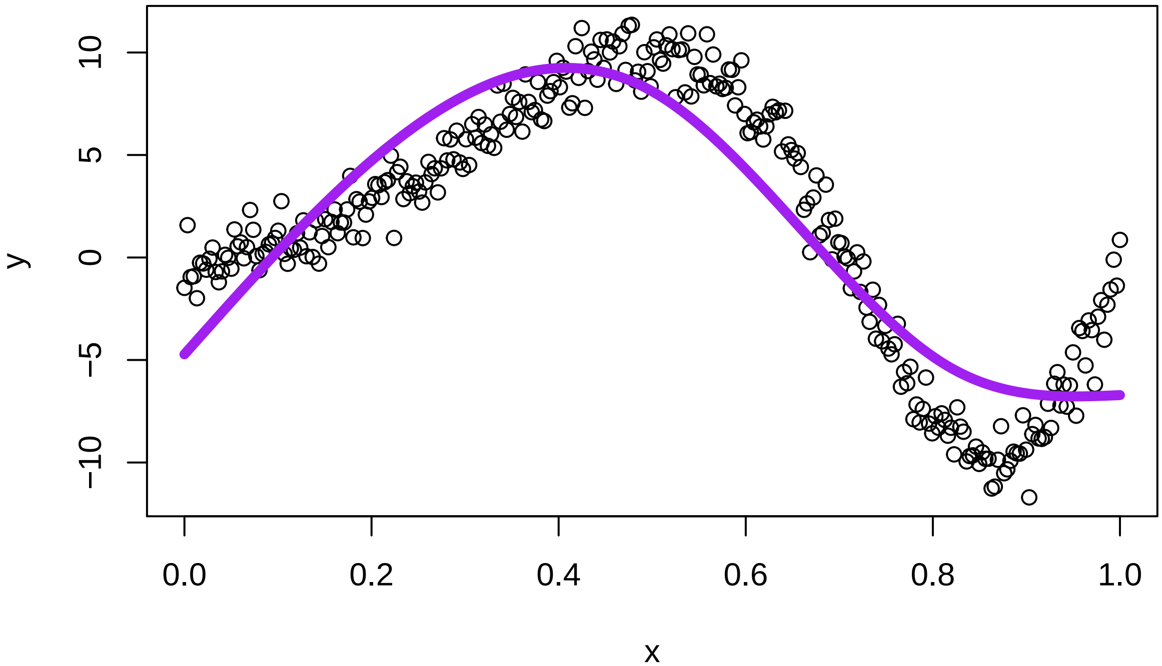
Too Many Knots

```
m2 <- lm(y ~ bs(x, knots = seq(0.1,1,by=0.02)), data = df)
pred <- predict(m2)

plot(x,y)
lines(x, pred, lwd = 5, col = "green")
```



Natural Splines

```r
m3 <- lm(y ~ ns(x, knots = c(0.5, 0.82)), data = df)
pred <- predict(m3)

plot(x,y)
lines(x, pred, lwd = 5, col = "purple")
```




```r
summary(m1)
```

```
##
## Call:
## lm(formula = y ~ bs(x, knots = c(0.525, 0.865)), data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.8195 -0.9079  0.0429  0.8680  3.2511
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       1.3218     0.3451   3.830 0.000157 ***
## bs(x, knots = c(0.525, 0.865))1  -7.0653     0.7472  -9.455  < 2e-16 ***
## bs(x, knots = c(0.525, 0.865))2  25.4800     0.5165  49.333  < 2e-16 ***
## bs(x, knots = c(0.525, 0.865))3 -12.6591     0.6171 -20.514  < 2e-16 ***
## bs(x, knots = c(0.525, 0.865))4 -11.0100     0.5118 -21.512  < 2e-16 ***
## bs(x, knots = c(0.525, 0.865))5   0.1012     0.6752   0.150 0.880987
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.269 on 294 degrees of freedom
## Multiple R-squared:  0.9594, Adjusted R-squared:  0.9587
## F-statistic:  1390 on 5 and 294 DF,  p-value: < 2.2e-16
```