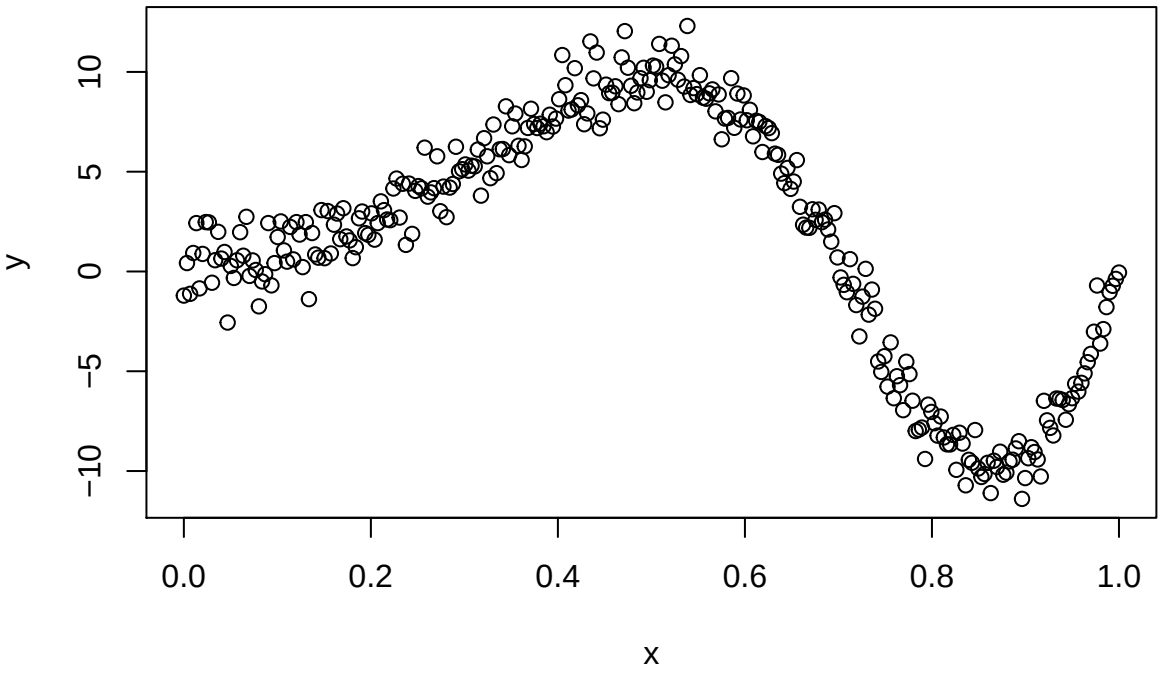
Introduction to Splines

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+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

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

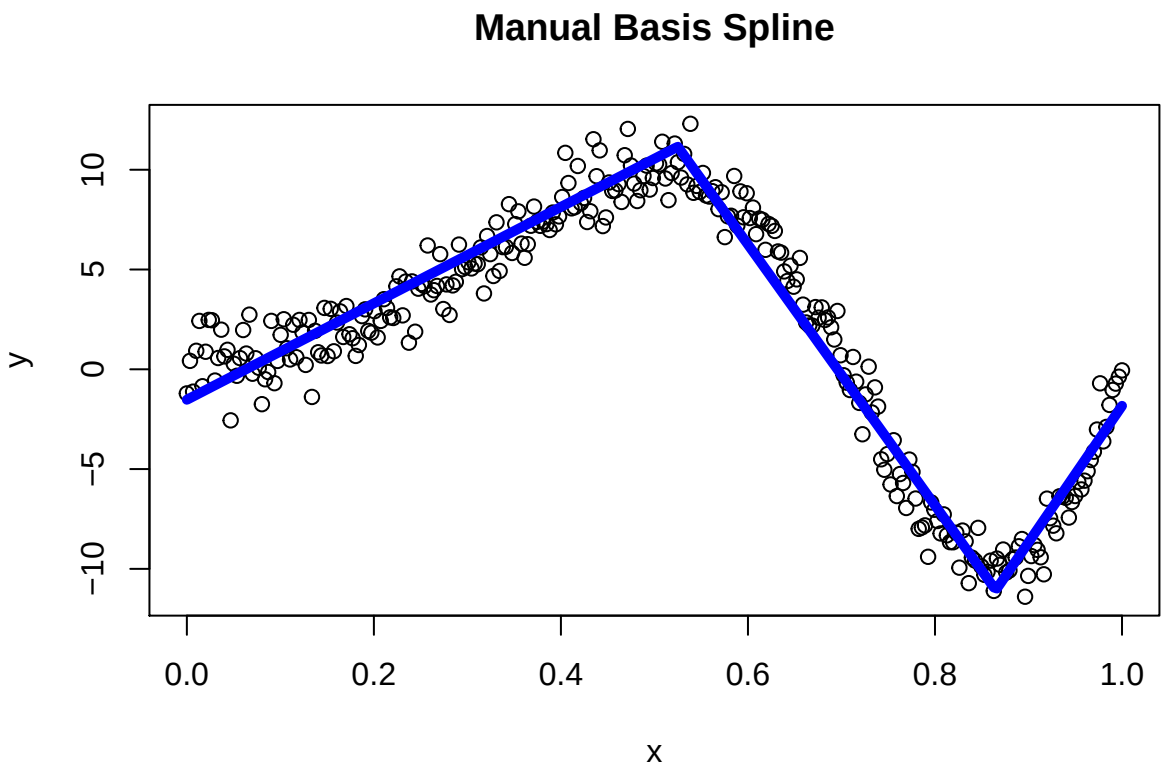
where $s \in \{-1, 1\}$, which we'll call the sign, and t is a value in the domain of x, which we will call a knot. Also, $[a]_+ = max(0, a)$.

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.

Manual Spline Function

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

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

  mBmat <- t(Bmat)
  mod <- lm(y ~ x + mBmat)
  pred <- predict(mod)

  sq <- x
  for(ii in 1:nknot) {
    hs[ii,] <- 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")
}
```

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

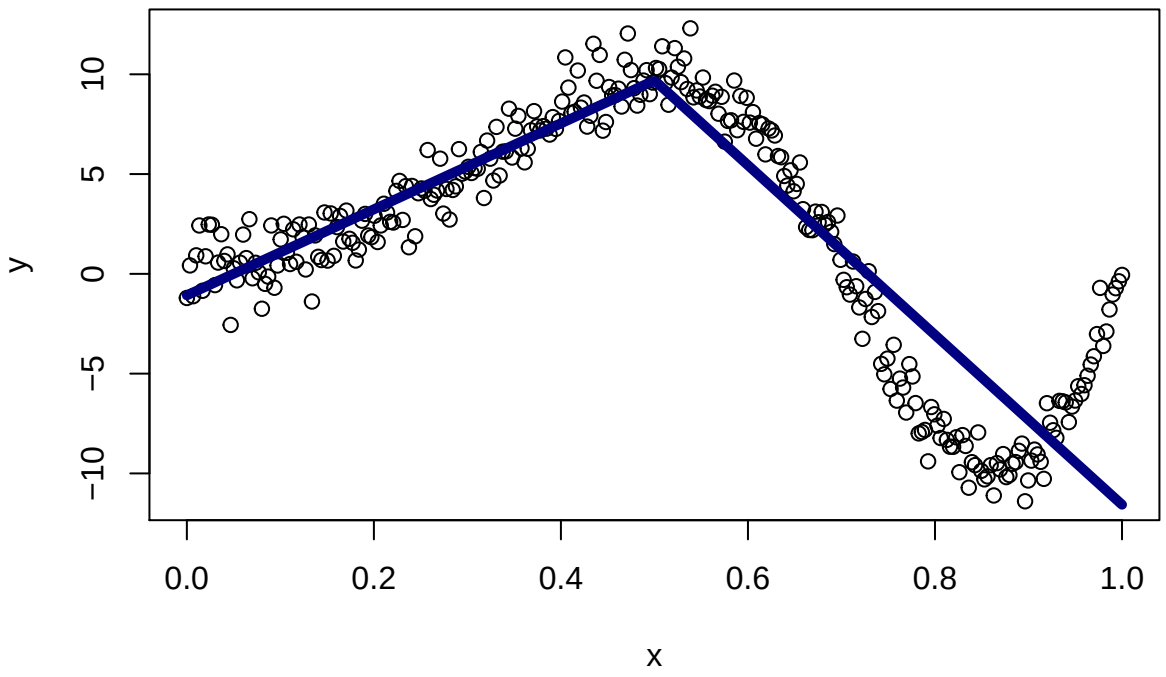


Trying things out

```
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)
```

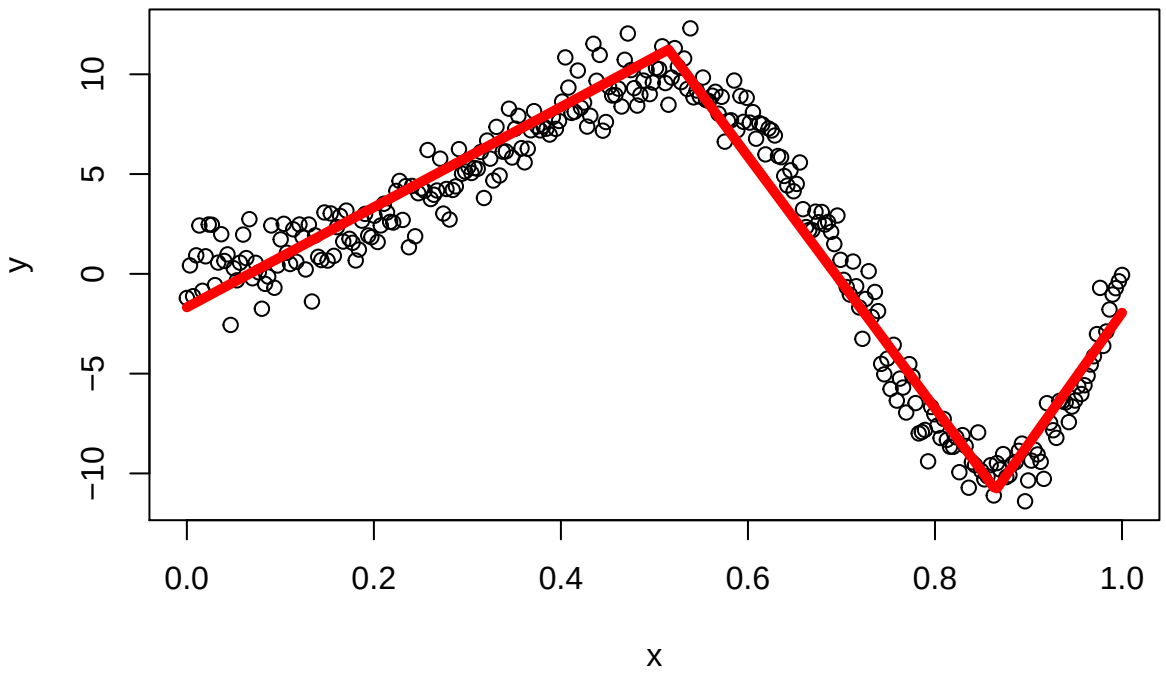
```
##
## Call:
## lm(formula = y ~ x + B1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.6472 -1.3335  0.0453  1.3109 11.5126
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.0752     0.4286  -2.508   0.0127 *
## x            21.5431     1.2561  17.150   <2e-16 ***
## B1          -64.0698     2.2471 -28.513   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.818 on 297 degrees of freedom
## Multiple R-squared:  0.7964, Adjusted R-squared:  0.795
## F-statistic: 580.9 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 <- 0.515
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 ~ x + 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")
```



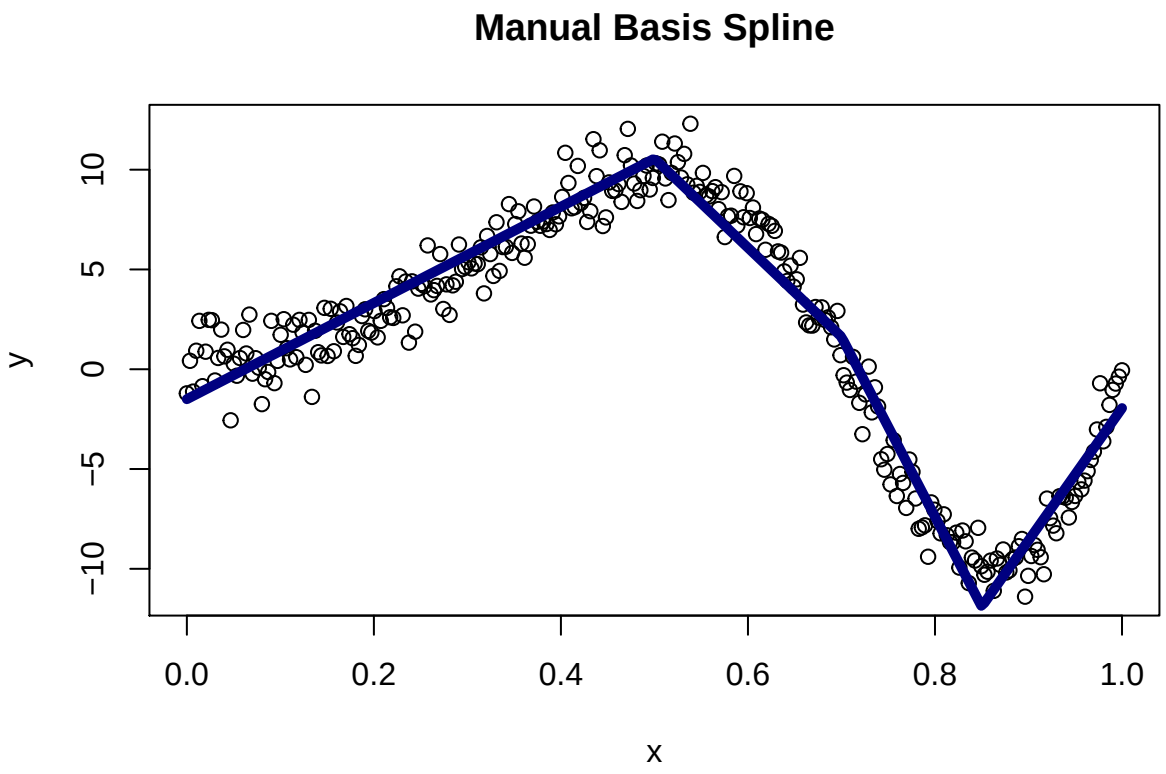
Add another knot:

```
t1 <- 0.5 #knot at 0.5
t2 <- 0.85 #another knot at 0.85
t3 <- 0.7
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 ~ x + B1 + B2 + B3)
summary(mod)
```

```
##
## Call:
## lm(formula = y ~ x + B1 + B2 + B3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.1121 -0.9573 -0.0679  0.8211  3.6056
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.4997     0.2102  -7.135 7.52e-12 ***
## x            24.1074     0.6648  36.262  < 2e-16 ***
```

```
## B1          -68.7576     2.1312 -32.262  < 2e-16 ***
## B2          156.4738     5.6280  27.803  < 2e-16 ***
## B3          -45.4978     4.0024 -11.368  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.346 on 295 degrees of freedom
## Multiple R-squared:  0.9538, Adjusted R-squared:  0.9532
## F-statistic:  1524 on 4 and 295 DF,  p-value: < 2.2e-16
```

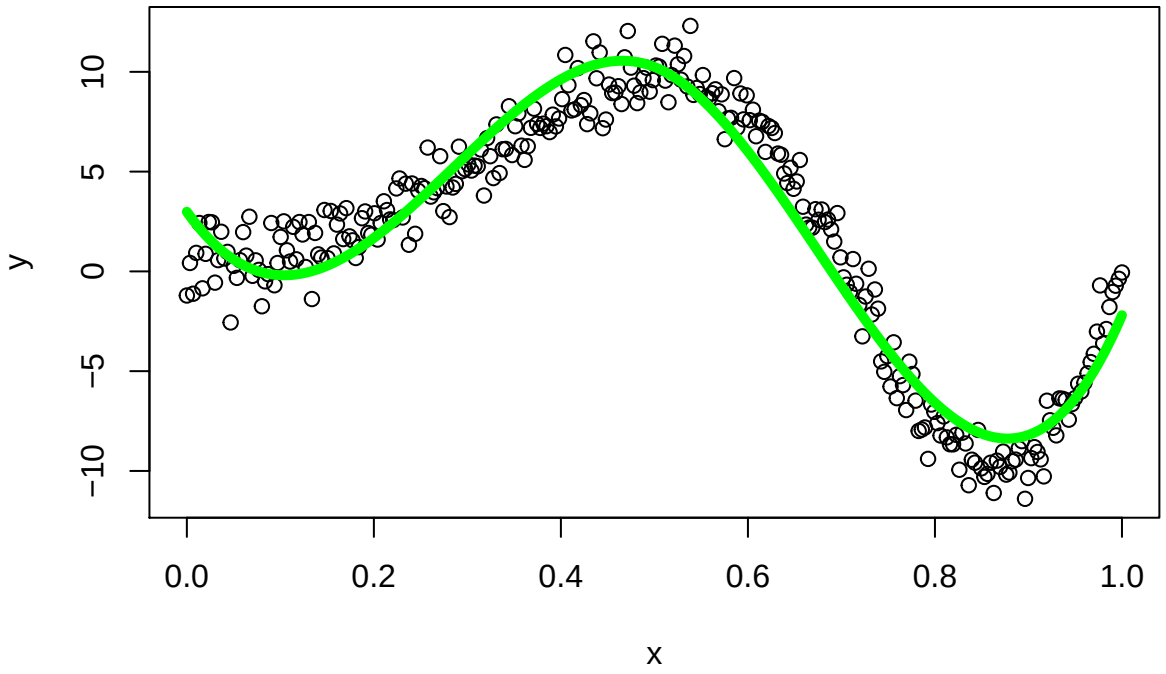
```r
cf <- mod$coefficients
sq <- x

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")
```



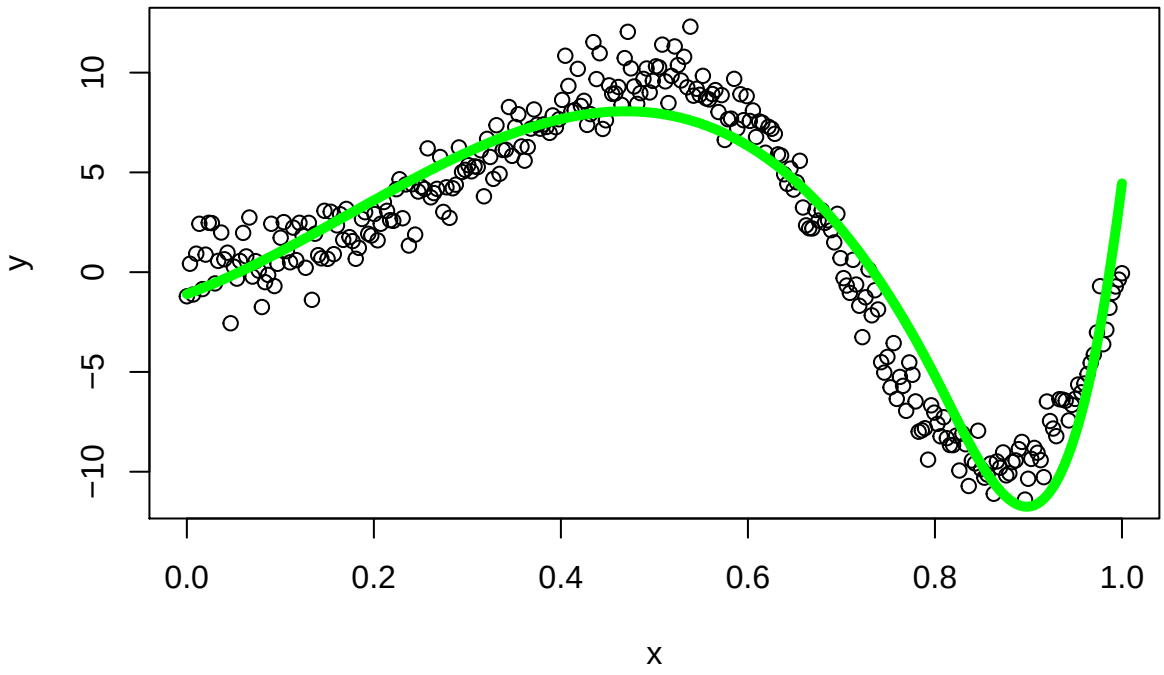
Using the `bs()` Function

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")
```

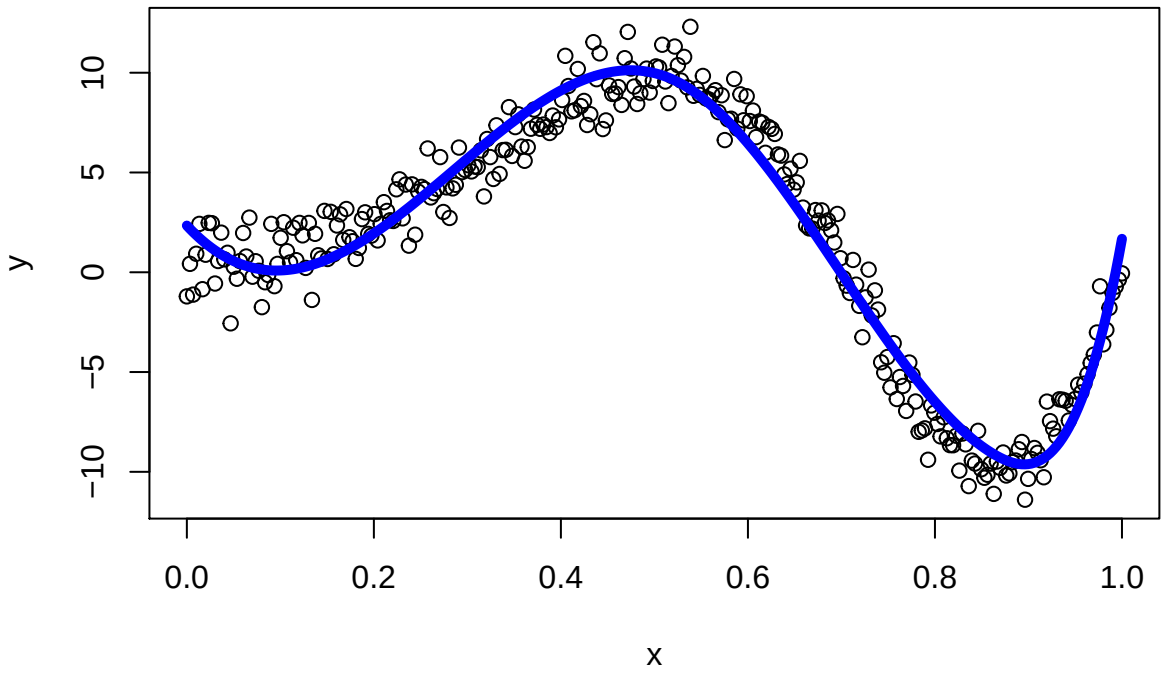



```
m2 <- lm(y ~ bs(x, knots = 0.8), data = df)
pred <- predict(m2)

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



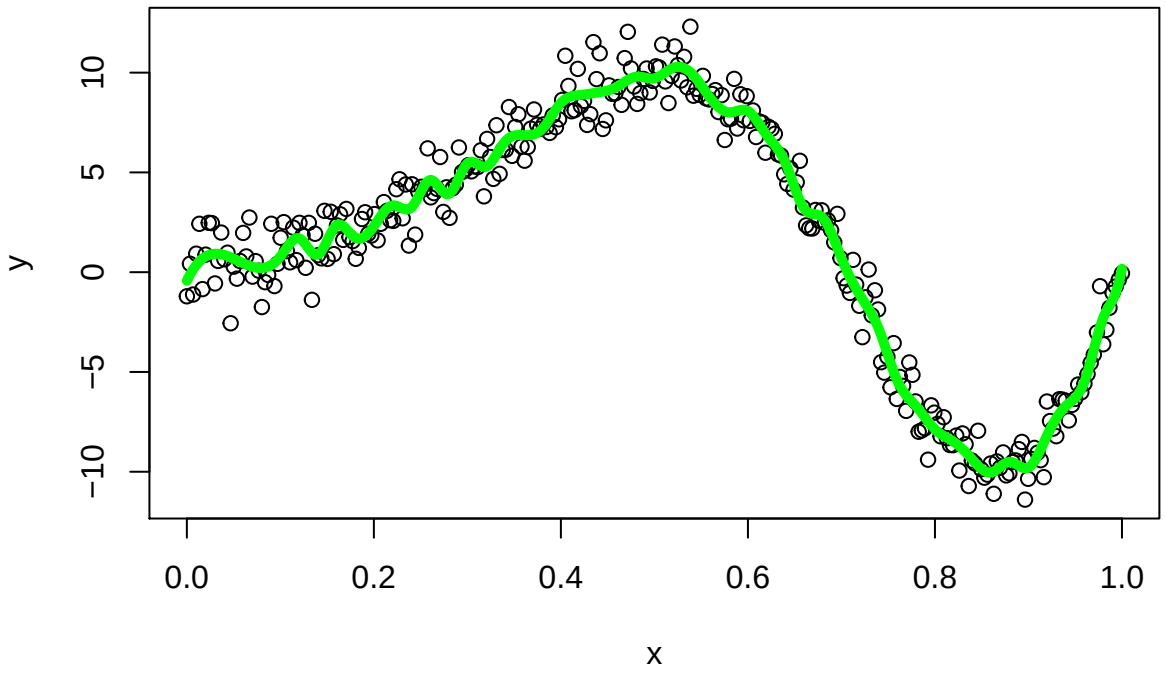
2 Knots (Expected)

```
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")
```



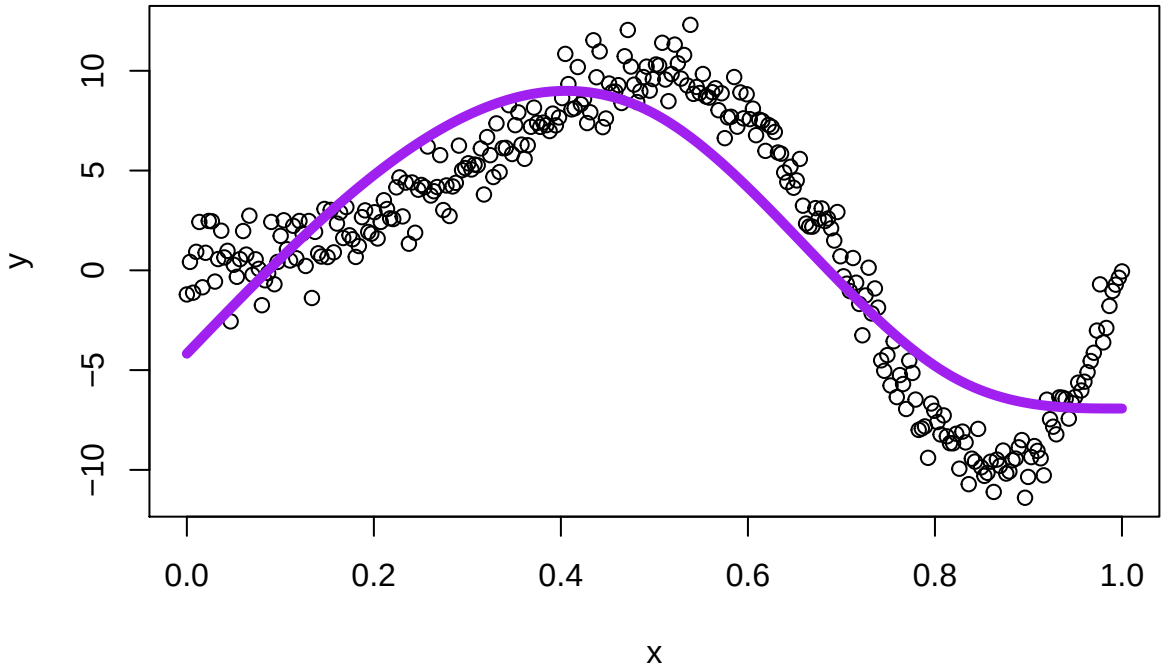
Too Many Knots

```
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

```
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")
```




```
summary(m1)
```

```
##
## Call:
## lm(formula = y ~ bs(x, knots = c(0.525, 0.865)), data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5480 -0.9593  0.0412  0.9431  3.1326
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       2.3390     0.3616   6.468 4.14e-10 ***
## bs(x, knots = c(0.525, 0.865))1  -8.8239     0.7830 -11.269  < 2e-16 ***
## bs(x, knots = c(0.525, 0.865))2  24.5124     0.5412  45.292  < 2e-16 ***
## bs(x, knots = c(0.525, 0.865))3 -13.6536     0.6466 -21.115  < 2e-16 ***
## bs(x, knots = c(0.525, 0.865))4 -12.3256     0.5363 -22.982  < 2e-16 ***
## bs(x, knots = c(0.525, 0.865))5  -0.6638     0.7075  -0.938    0.349
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.33 on 294 degrees of freedom
## Multiple R-squared:  0.9551, Adjusted R-squared:  0.9544
## F-statistic:  1251 on 5 and 294 DF,  p-value: < 2.2e-16
```