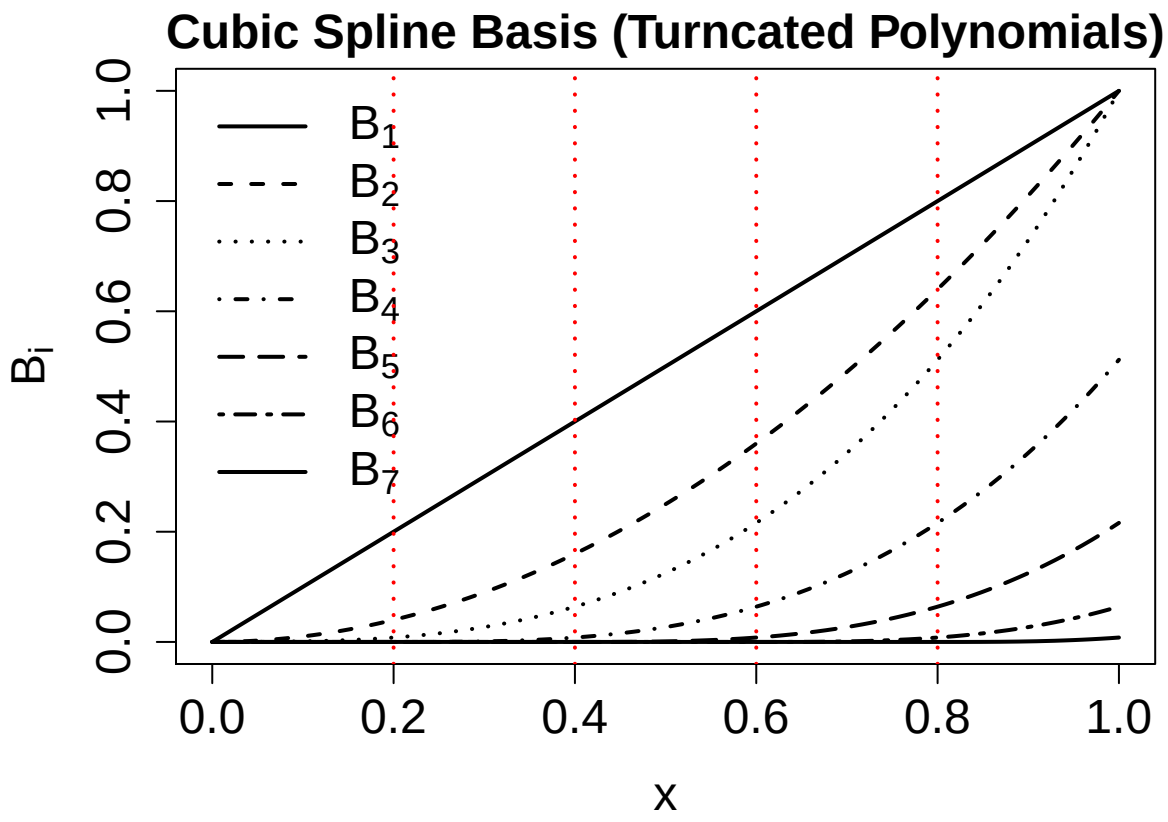
```
x = seq(0,1,length=100)
knots = c(0.2 , 0.4 , 0.6 , 0.8)

# Defining cubic splines basis functions
u = function(x,a){
  temp = x-a
  sapply(temp , max , 0)
}

B1=x
B2=x^2
B3=x^3
B4=(u(x,0.2))^3
B5=(u(x,0.4))^3
B6=(u(x,0.6))^3
B7=(u(x,0.8))^3

# Plotting cubic splines basis functions
par(mar=c(5,5,2,2))
plot(B1~x,type="l" , ylab = expression(B[i]),
     cex.lab = 1.5 , cex.axis=1.5 , cex.main = 1.5, lty=1, col=1,lwd=2,
     main = "Cubic Spline Basis (Turncated Polynomials)")
lines(B2~x , lty=2 , col=1,lwd=2)
lines(B3~x, lty=3, col=1,lwd=2)
lines(B4~x, lty=4, col=1,lwd=2)
lines(B5~x, lty=5, col=1,lwd=2)
lines(B6~x, lty=6, col=1,lwd=2)
lines(B7~x, lty=7, col=1,lwd=2)
legend("topleft", bty="n", cex=1.5,
       legend = c(expression(B[1]), expression(B[2]),expression(B[3]),
                  expression(B[4]), expression(B[5]) , expression(B[6]),
                  expression(B[7])), lwd=2, lty = 1:7, text.width = 2)
abline(v=knots , lty=3, lwd=2 , col="red")
```

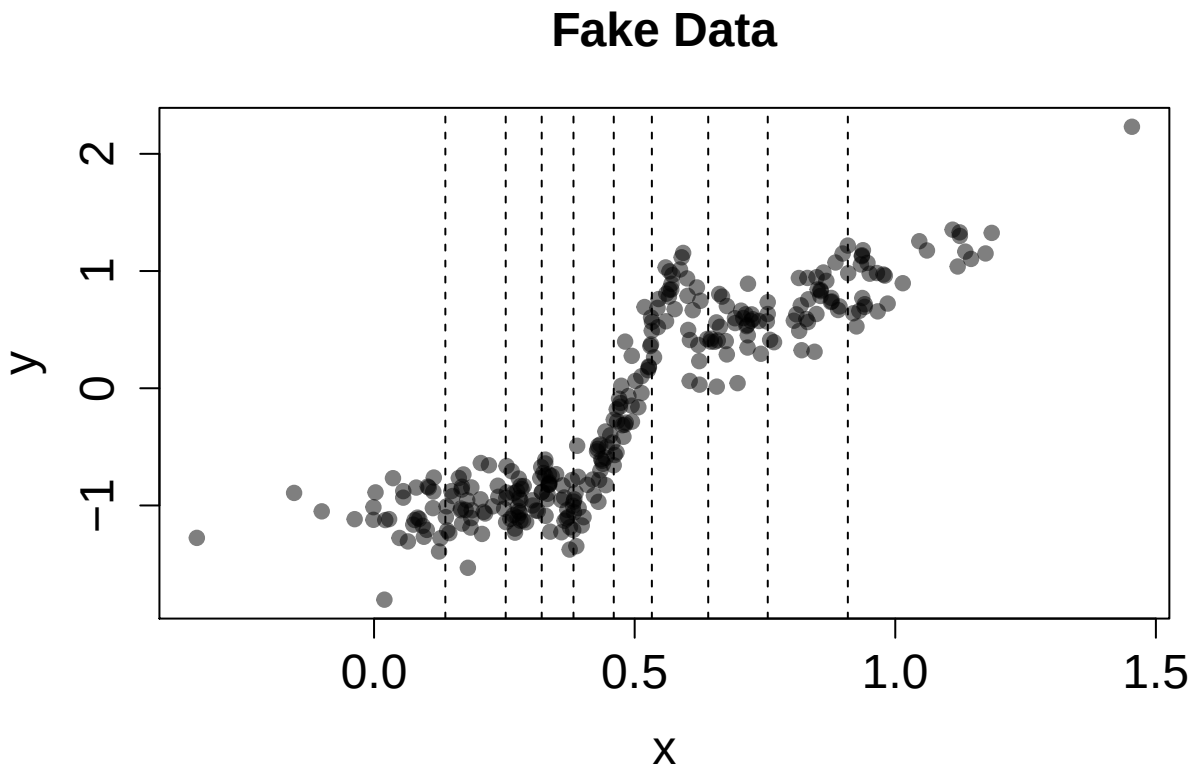


Fake Data Example.

```
# Get some x's
#
set.seed(444)
N <- 300
x <- runif(150, 0,1)
x <- c(x, rnorm(150, 0.5, 0.3))
#
#  A function for the mean of ys
#
mu <- function(x)
{
    xrange <- range(x)
    vals <- vector("numeric", length=length(x))
    breaks <- quantile(x, probs=c(1/3,2/3))
    first <- x <= breaks[1]
    second <- (x > breaks[1]) & (x <= breaks[2])
    third  <- x > breaks[2]
    vals[first]  <- -(1 - x[first])^0.5 -0.1
    vals[second] <- sin(x[second] * 4 * pi) + x[second]/10
    vals[third]  <- x[third]^2
    vals
}
```

```
#
# the mean function
#
# xordered <- sort(x)
# plot(xordered, mu(xordered), type="l", lwd=2, lty=2)
#
# generate some ys
#
y <- mu(x) + rnorm(N, 0, .2)

plot(x,y,
     pch=19, cex=1,
     main = "Fake Data",
     cex.axis=1.5 , cex.main=1.5, cex.lab=1.5,
     col=adjustcolor("black" , alpha=0.5)
)
knots_p <- quantile(x, seq(0.1, 0.9, 0.1))
abline(v=knots_p , lty=2 )
```



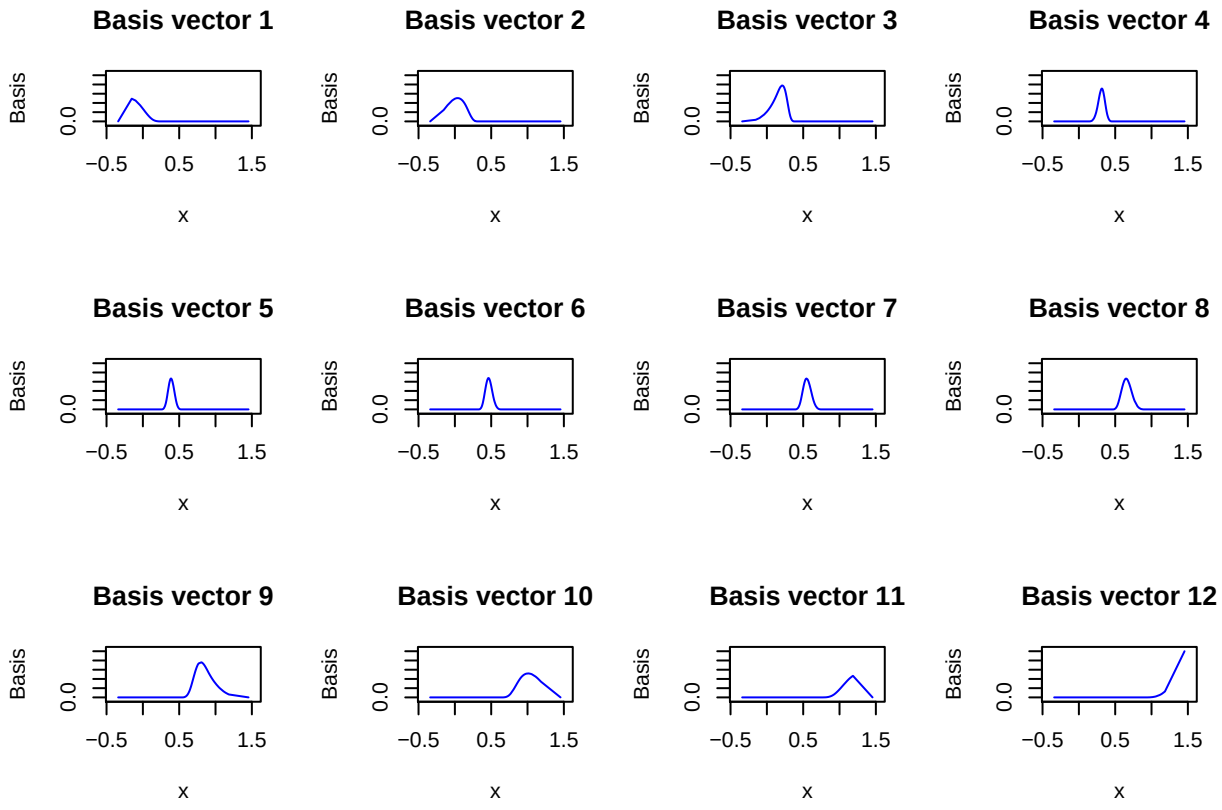
B-Splines.

```
library(splines)
p <- 3
# Note that the knots here are the interior knots
# To match our previous fits, we might choose
```

```
Xmat <- bs(x, degree= p, knots=knots_p)
#
# This will be an N by (p + length(knots_p)) matrix
# the first few rows of which are
head(Xmat)
```

```
##                1         2           3           4         5          6
## [1,] 0.014856148 0.2866634 0.692679285 0.005801203 0.0000000 0.00000000
## [2,] 0.007557959 0.2289939 0.742031661 0.021416499 0.0000000 0.00000000
## [3,] 0.000000000 0.0000000 0.000000000 0.156418026 0.6674475 0.17601923
## [4,] 0.000000000 0.0000000 0.002596317 0.369311095 0.5808317 0.04726084
## [5,] 0.175836957 0.5033455 0.314544910 0.000000000 0.0000000 0.00000000
## [6,] 0.000000000 0.0000000 0.000000000 0.000000000 0.0000000 0.05254434
##                 7         8           9 10 11 12
## [1,] 0.0000000000 0.0000000 0.000000000  0  0  0
## [2,] 0.0000000000 0.0000000 0.000000000  0  0  0
## [3,] 0.0001152409 0.0000000 0.000000000  0  0  0
## [4,] 0.0000000000 0.0000000 0.000000000  0  0  0
## [5,] 0.0000000000 0.0000000 0.000000000  0  0  0
## [6,] 0.5871495451 0.3510166 0.009289472  0  0  0
```

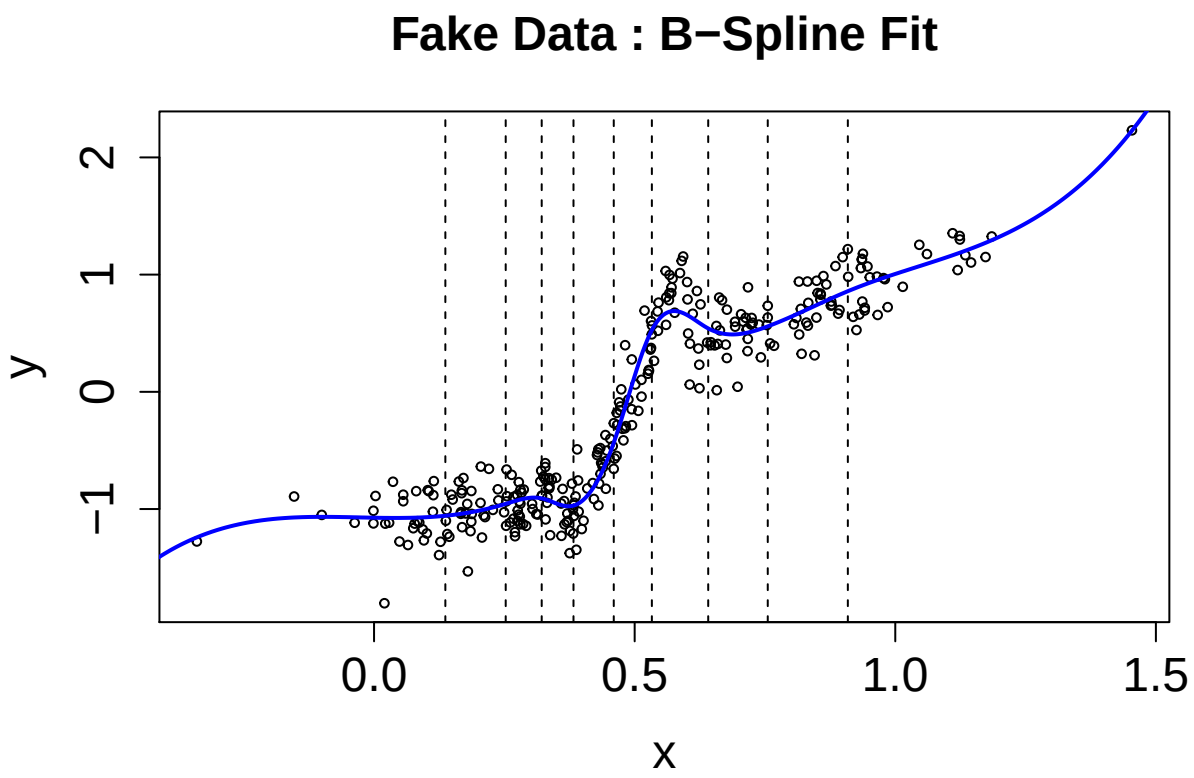
```
Xorder <- order(x)
blim <- extendrange(Xmat)
parOptions <- par(mfrow = c(3,4))
for (j in 1:ncol(Xmat))
{
    plot(x[Xorder], Xmat[Xorder,j],
         type="l",
         ylim=blim,
         xlim = extendrange(x),
         xlab="x", ylab="Basis",
         main=paste("Basis vector", j),
         col="blue")
}
```




```
# Fit of the B-spline model
fit.bs <- lm(y ~ bs(x, degree= p, knots=knots_p))
# Let's get predictions at a lot of x-values equi-spaced
# across the range of x so that we can see how smooth
# the fit is
xrange <- extendrange(x)
xnew <- seq(min(xrange), max(xrange), length.out=500)
ypred.bs <- predict(fit.bs, newdata= data.frame(x=xnew))
```

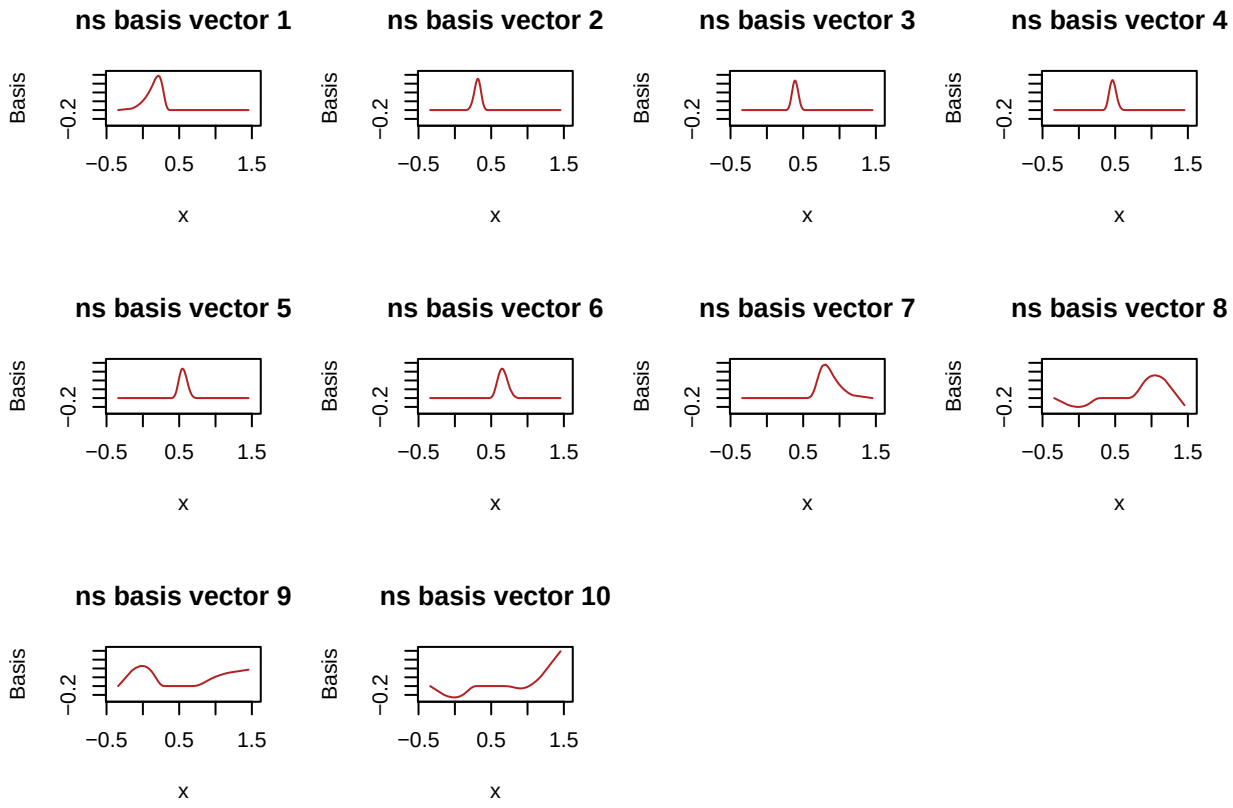
```
## Warning in bs(x, degree = 3L, knots = c(`10%` = 0.136800685874186, `20%` =
## 0.252579851988159, : some 'x' values beyond boundary knots may cause ill-
## conditioned bases
```

```
par(mfrow=c(1,1))
plot(x,y,
     pch=1, cex=0.6,
     main = "Fake Data : B-Spline Fit",
     cex.axis=1.5 , cex.main=1.5, cex.lab=1.5,
)
abline(v=knots_p , lty=2 )
lines(xnew, ypred.bs, col="blue", lwd=2)
```

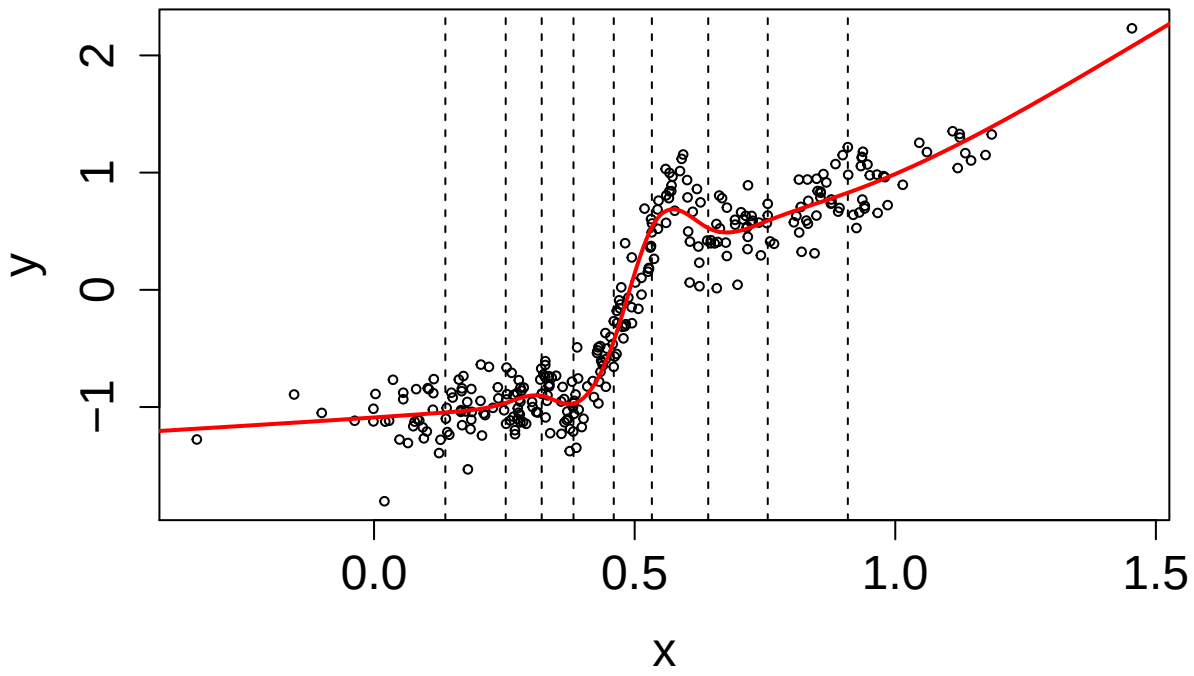


Natural Splines.

```
par(mfrow = c(3,4))
Xmat.ns <- ns(x, knots=knots_p)
blim <- extendrange(Xmat.ns)
parOptions <- par(mfrow = c(3,4))
for (j in 1:ncol(Xmat.ns))
{
    plot(x[Xorder], Xmat.ns[Xorder,j],
         type="l",
         ylim=blim,
         xlim = extendrange(x),
         xlab="x", ylab="Basis",
         main=paste("ns basis vector", j),
         col="firebrick")
}
```




```r
# Fit of the natural spline model
par(mfrow=c(1,1))
plot(x,y,
     pch=1, cex=0.6,
     #main = "Fake Data : B-Spline Fit",
     cex.axis=1.5 , cex.main=1.5, cex.lab=1.5,
)
abline(v=knots_p , lty=2 )
fit.ns <- lm(y ~ ns(x, knots=knots_p))
ypred.ns <- predict(fit.ns, newdata= data.frame(x=xnew))
lines(xnew, ypred.ns, col="red",lwd=2)
```



Varying degrees of freedom

```
fit1 <- lm(y ~ bs(x, degree= 3, df=4))
fit2 <- lm(y ~ bs(x, degree= 3, df=5))
fit3 <- lm(y ~ bs(x, degree= 3, df=8))
ypred1 <- predict(fit1,newdata=data.frame(x=xnew))
```

```
## Warning in bs(x, degree = 3L, knots = c(`50%` = 0.459919521008099),
## Boundary.knots = c(-0.339967486304014, : some 'x' values beyond boundary knots
## may cause ill-conditioned bases
```

```
ypred2 <- predict(fit2,newdata=data.frame(x=xnew))
```

```
## Warning in bs(x, degree = 3L, knots = c(`33.33333%` = 0.335309901549353, : some
## 'x' values beyond boundary knots may cause ill-conditioned bases
```

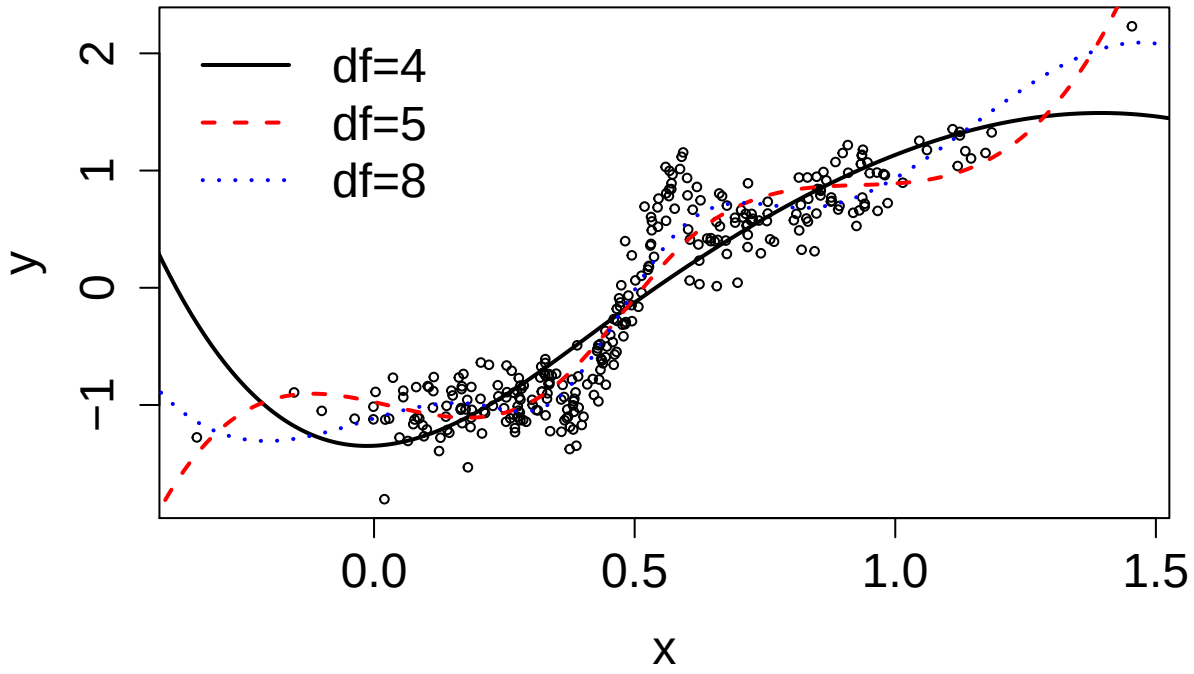
```
ypred3 <- predict(fit3,newdata=data.frame(x=xnew))
```

```
## Warning in bs(x, degree = 3L, knots = c(`16.66667%` = 0.201501496407824, : some
## 'x' values beyond boundary knots may cause ill-conditioned bases
```

```
plot(x,y,
     pch=1, cex=0.6,
     #main = "Fake Data : B-Spline Fit",
     cex.axis=1.5 , cex.main=1.5, cex.lab=1.5,
)
lines(xnew, ypred1, col="black", lwd=2, lty=1)
```

```
lines(xnew, ypred2, col="red", lwd=2, lty=2)
lines(xnew, ypred3, col="blue", lwd=2, lty=3)

legend("topleft", bty="n", cex=1.5,
       legend=c("df=4","df=5", "df=8"),
       lty=c(1,2,3), lwd=2,
       col=c("black", "red", "blue")
)
```




```
# More knots where there is variability
knots_p2 <- quantile(x,
                     c(seq(0.1,0.6,0.1),
                       0.63, 0.65,
                       seq(0.7,0.9,0.1)
                     )
)

fit.ns2 <- lm(y ~ ns(x, knots=knots_p2))

# And now the predicted values for plotting
ypred.ns2 <- predict(fit.ns2, newdata= data.frame(x=xnew))


plot(x,y,
     pch=1, cex=0.6,
     main = "Comparing cubic spline, knots at lines",
```

```
     cex.axis=1.5 , cex.main=1.5, cex.lab=1.5,
)
abline(v=knots_p, col ="grey", lty=2)
abline(v=knots_p2[c(7,8)], col ="grey30", lty=2)
lines(xnew, ypred.ns, col="blue", lwd=2, lty=1)
lines(xnew, ypred.ns2, col="red", lwd=2, lty=1)
legend("topleft", bty="n", cex=1.5,
       legend=c("ns - natural spline","ns - extra knots"),
       lty=c(1,1), lwd=2,
       col=c("blue", "red")
)
```