Gam fitting and diagnostics

John McKinlay

Tuesday, February 17, 2015

Simulate some data, fit a model. Fit a second model with >> k. Compare.

```
# gam diagnostics

library(mgcv)
```

```
## Loading required package: nlme
## This is mgcv 1.8-3. For overview type 'help("mgcv-package")'.
```

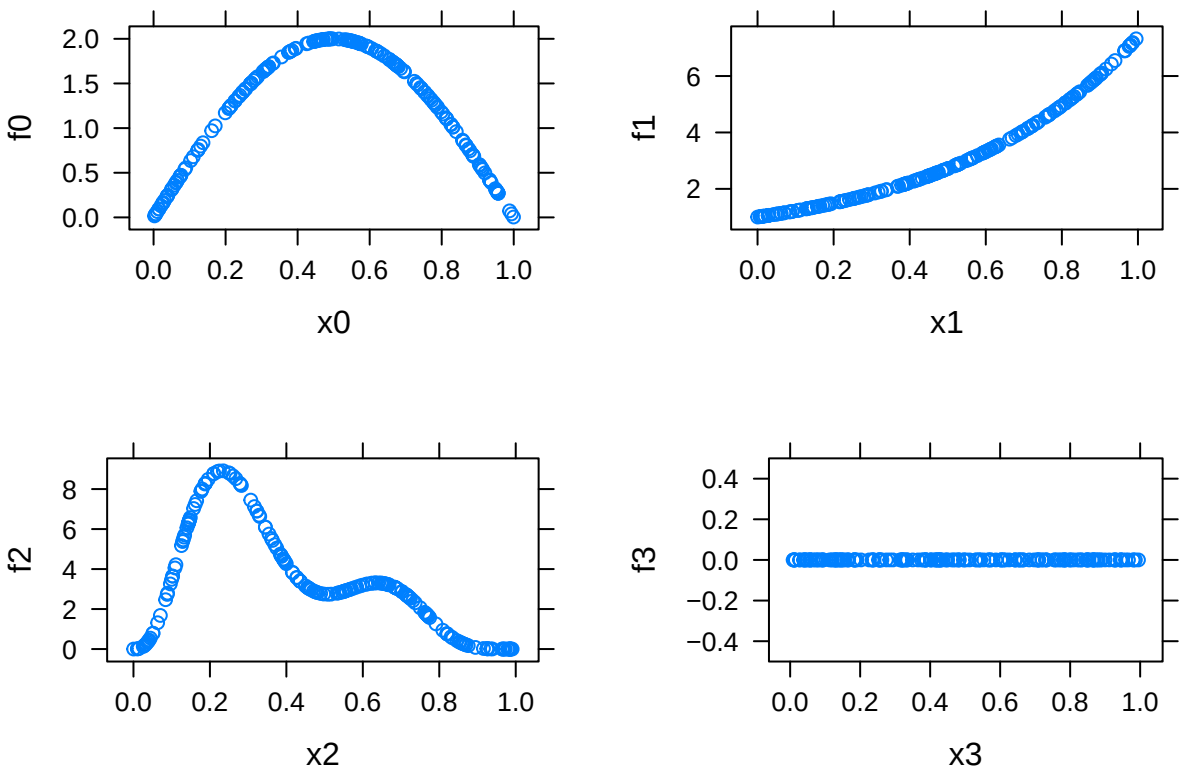
```
dat <- gamSim(1,n=200)
```

```
## Gu & Wahba 4 term additive model
```

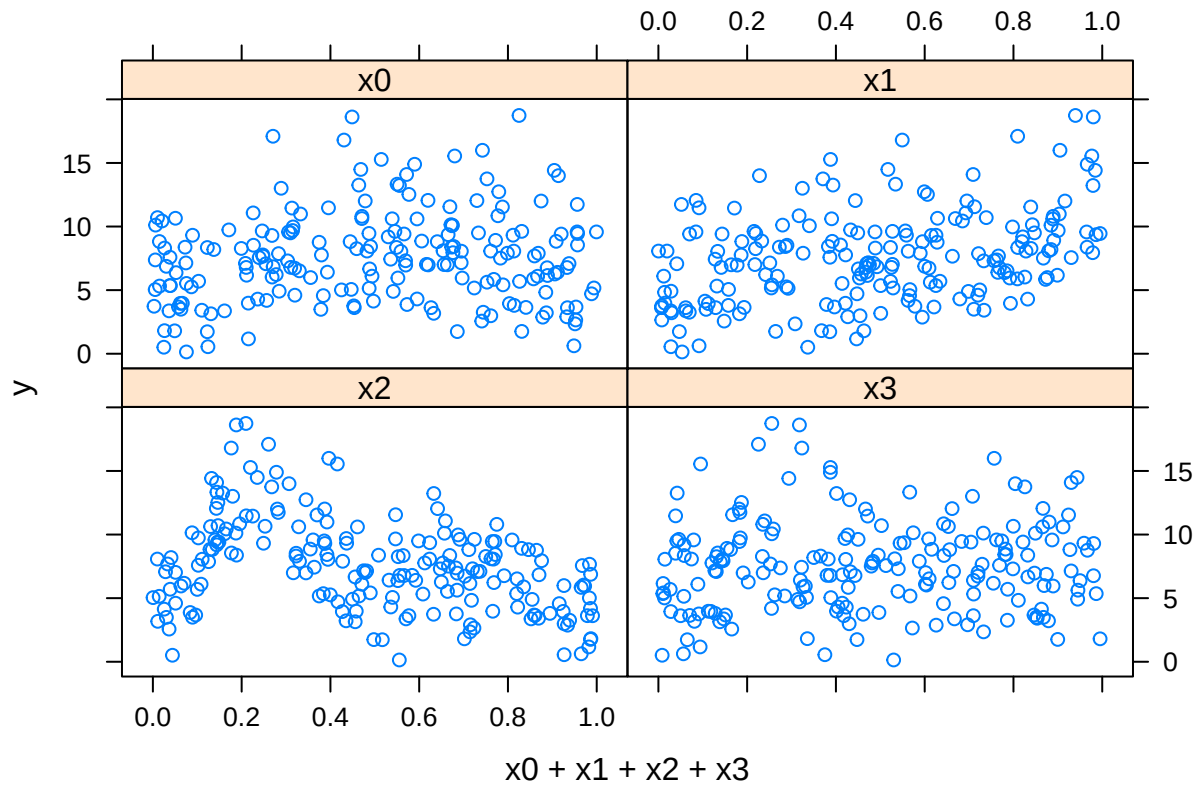
```
head(dat)
```

```
##       y     x0     x1     x2      x3     f     f0    f1        f2 f3
## 1 8.915 0.7721 0.8907 0.8308 0.15462 7.862 1.3126 5.939 6.103e-01  0
## 2 3.622 0.6267 0.1290 0.9789 0.70964 3.138 1.8437 1.294 1.380e-05  0
## 3 6.029 0.2687 0.8256 0.9704 0.60403 6.708 1.4947 5.213 9.582e-05  0
## 4 5.169 0.7241 0.5287 0.3748 0.56694 9.452 1.5244 2.879 5.049e+00  0
## 5 5.415 0.7888 0.2558 0.4898 0.01141 5.677 1.2318 1.668 2.777e+00  0
## 6 2.875 0.8784 0.5945 0.9344 0.62592 4.037 0.7457 3.284 7.576e-03  0
```

```
library(lattice)
print(xyplot(f0~x0, data=dat), split=c(1,1,2,2), more=TRUE)
print(xyplot(f1~x1, data=dat), split=c(2,1,2,2), more=TRUE)
print(xyplot(f2~x2, data=dat), split=c(1,2,2,2), more=TRUE)
print(xyplot(f3~x3, data=dat), split=c(2,2,2,2), more=FALSE)
```




```
xyplot(y~x0+x1+x2+x3, data=dat, outer=TRUE, as.table=TRUE)
```




```
b <- gam(y~s(x0)+s(x1)+s(x2)+s(x3), data=dat)
b
```

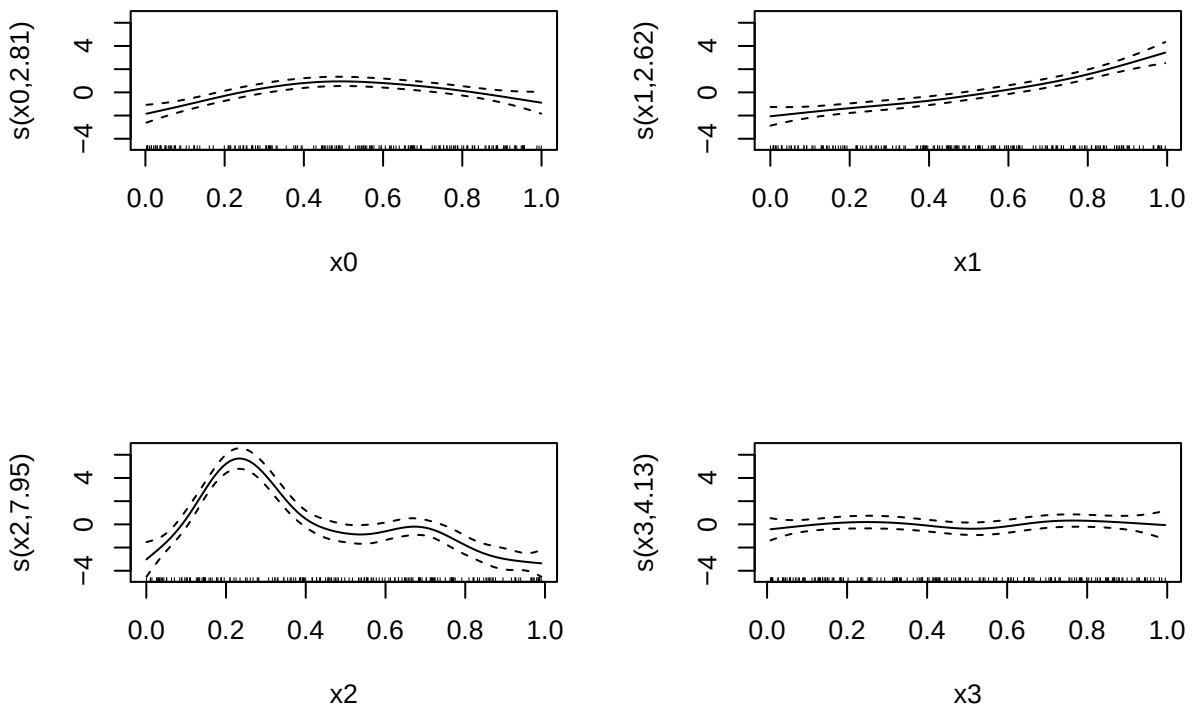
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## y ~ s(x0) + s(x1) + s(x2) + s(x3)
##
## Estimated degrees of freedom:
## 2.81 2.62 7.95 4.13  total = 18.51
##
## GCV score: 4.329
```

```
summary(b)
```

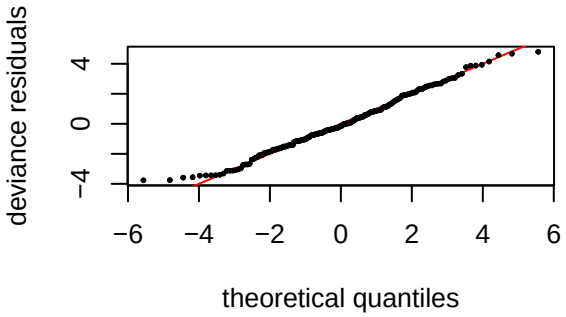
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## y ~ s(x0) + s(x1) + s(x2) + s(x3)
##
## Parametric coefficients:
```

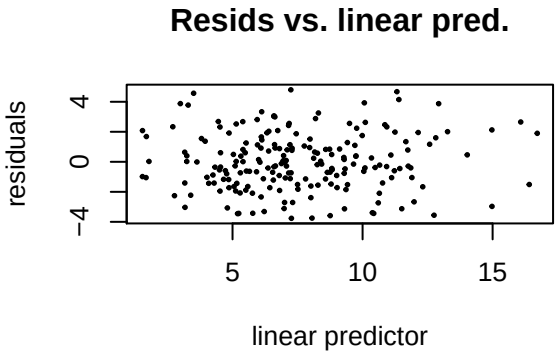
```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)     7.55       0.14    53.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##        edf Ref.df     F p-value
## s(x0) 2.81   3.46 10.55 6.7e-07 ***
## s(x1) 2.62   3.26 31.52 < 2e-16 ***
## s(x2) 7.95   8.70 32.95 < 2e-16 ***
## s(x3) 4.13   5.09  0.69    0.64
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.696   Deviance explained = 72.3%
## GCV = 4.3287  Scale est. = 3.9281    n = 200
```

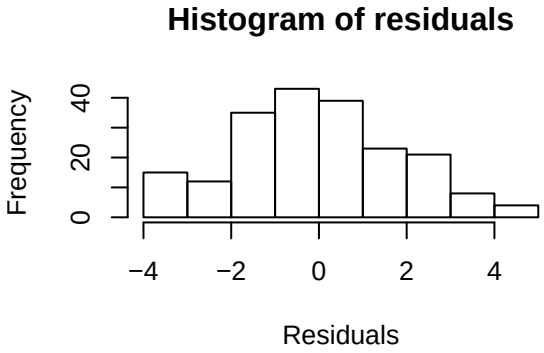
```
plot(b, pages=1)
```

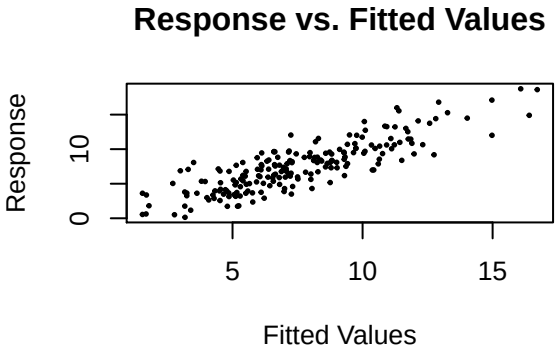



```
gam.check(b, pch=19, cex=.3)
```










```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 12 iterations.
## The RMS GCV score gradiant at convergence was 1.705e-06 .
## The Hessian was positive definite.
## The estimated model rank was 37 (maximum possible: 37)
## Model rank =  37 / 37
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##          k'   edf k-index p-value
## s(x0) 9.000 2.809   1.058    0.79
## s(x1) 9.000 2.624   1.101    0.92
## s(x2) 9.000 7.950   0.992    0.50
## s(x3) 9.000 4.126   1.052    0.78
```

```
b1 <- gam(y~s(x0, k=20)+s(x1, k=20)+s(x2, k=20)+s(x3, k=20), data=dat)
b1
```

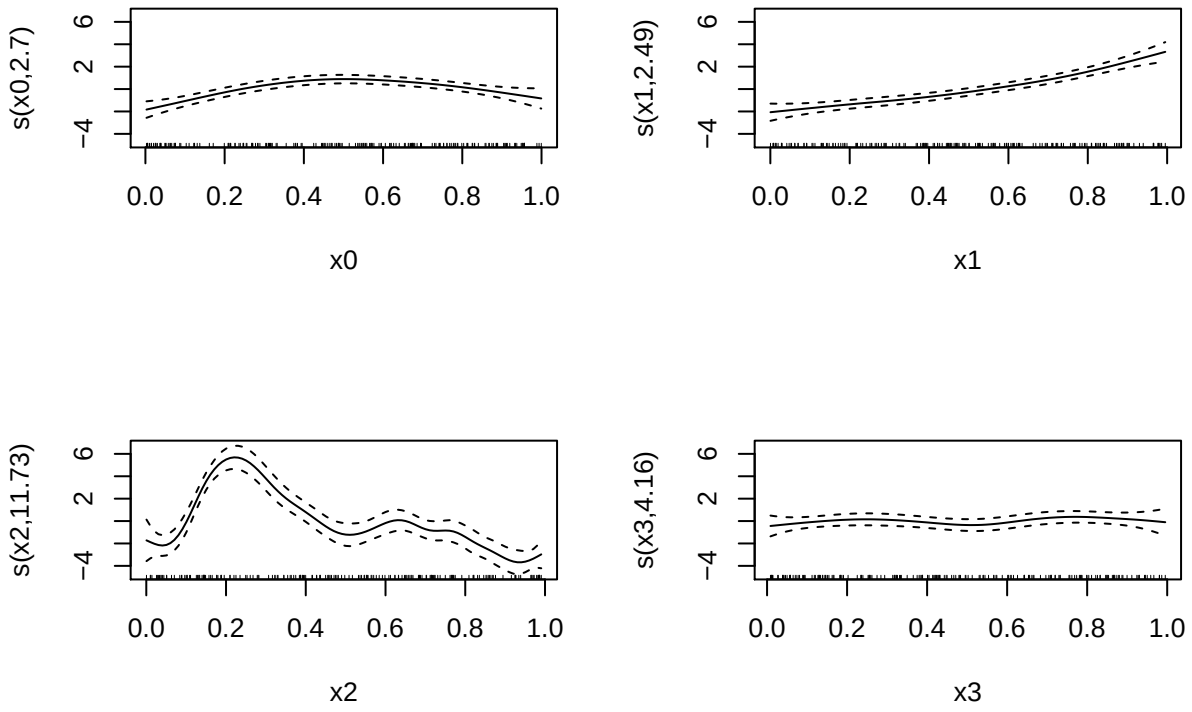
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
```

```
## y ~ s(x0, k = 20) + s(x1, k = 20) + s(x2, k = 20) + s(x3, k = 20)
##
## Estimated degrees of freedom:
##  2.70  2.49 11.73  4.16  total = 22.08
##
## GCV score: 4.139
```

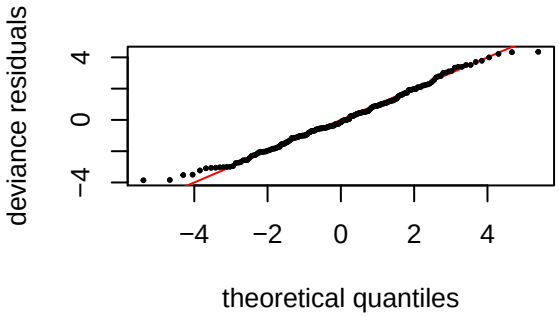
```
summary(b1)
```

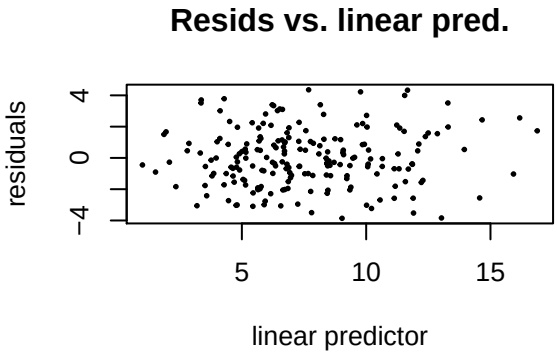
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## y ~ s(x0, k = 20) + s(x1, k = 20) + s(x2, k = 20) + s(x3, k = 20)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    7.552      0.136    55.7   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df     F p-value
## s(x0)  2.70   3.33 11.11 4.5e-07 ***
## s(x1)  2.49   3.10 33.73 < 2e-16 ***
## s(x2) 11.73  14.10 22.65 < 2e-16 ***
## s(x3)  4.16   5.18  0.77    0.58
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.715   Deviance explained = 74.6%
## GCV = 4.1386  Scale est. = 3.6817    n = 200
```

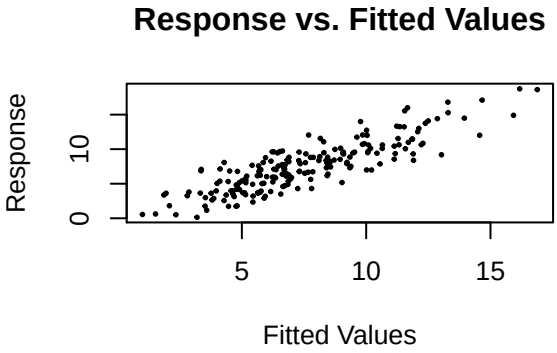
```
plot(b1, pages=1)
```




```
gam.check(b1, pch=19, cex=.3)
```








```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 10 iterations.
## The RMS GCV score gradiant at convergence was 1.546e-06 .
## The Hessian was positive definite.
## The estimated model rank was 77 (maximum possible: 77)
## Model rank =  77 / 77
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##          k'   edf k-index p-value
## s(x0) 19.00  2.70    1.08    0.88
## s(x1) 19.00  2.49    1.09    0.86
## s(x2) 19.00 11.73    1.09    0.88
## s(x3) 19.00  4.16    1.05    0.78
```

```
anova(b, b1, test="F")
```

```
## Analysis of Deviance Table
##
## Model 1: y ~ s(x0) + s(x1) + s(x2) + s(x3)
## Model 2: y ~ s(x0, k = 20) + s(x1, k = 20) + s(x2, k = 20) + s(x3, k = 20)
##   Resid. Df Resid. Dev   Df Deviance   F Pr(>F)
## 1       181        713
```

```
## 2       178        655 3.57     57.9 4.4  0.003 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
AIC(b)
```

```
## [1] 862.8
```

```
AIC(b1)
```

```
## [1] 853
```

```
BIC(b)
```

```
## [1] 927.1
```

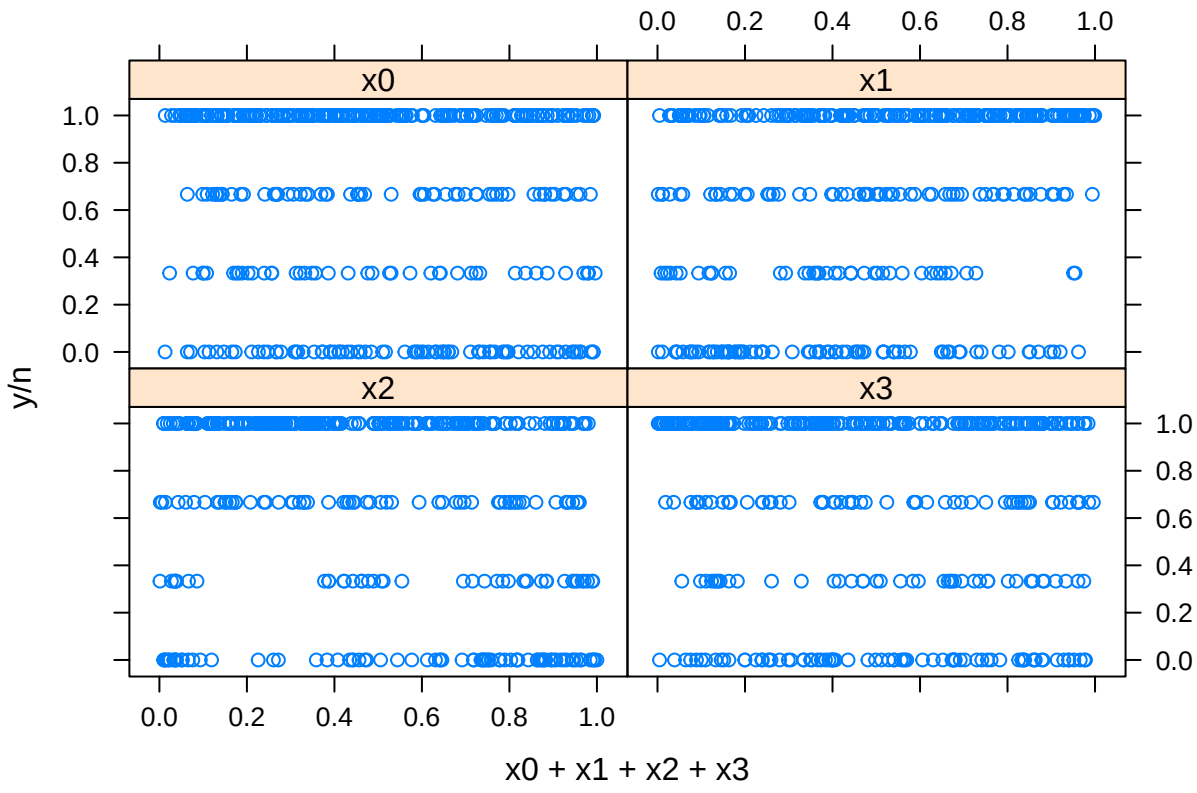
```
BIC(b1)
```

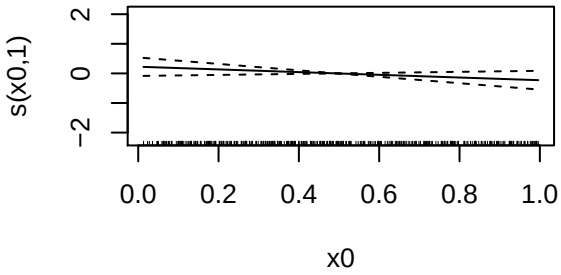
```
## [1] 929.2
```

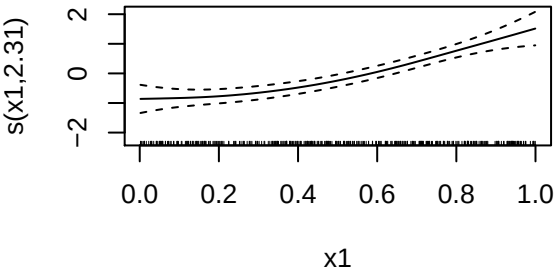
Simulate binomial data.

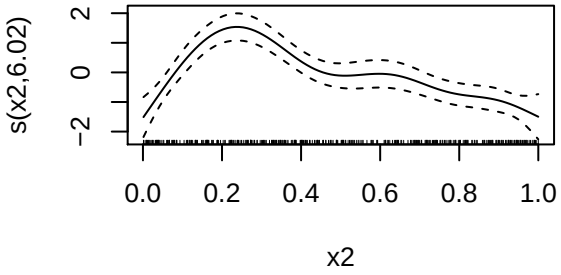
```
## Gu & Wahba 4 term additive model
```

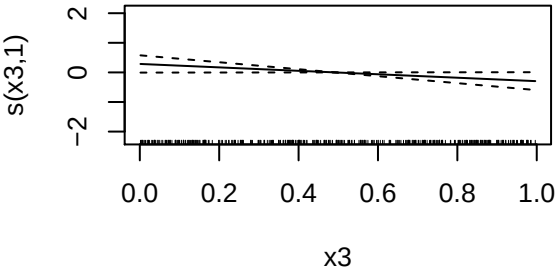
```
##   y     x0     x1      x2      x3       f     f0    f1        f2 f3 n
## 1 1 0.8967 0.1478 0.34826 0.04572  0.9776 0.6377 1.344 5.980e+00  0 1
## 2 3 0.2655 0.6589 0.85869 0.36653  0.1698 1.4814 3.735 2.981e-01  0 3
## 3 0 0.3721 0.1851 0.03444 0.74139 -0.4698 1.8408 1.448 2.877e-01  0 3
## 4 1 0.5729 0.9544 0.97100 0.93351  1.2186 1.9478 6.745 8.611e-05  0 3
## 5 0 0.9082 0.8978 0.74511 0.67321  1.2384 0.5688 6.024 2.160e+00  0 1
## 6 1 0.2017 0.9437 0.27326 0.70136  3.6928 1.1841 6.602 8.404e+00  0 1
```

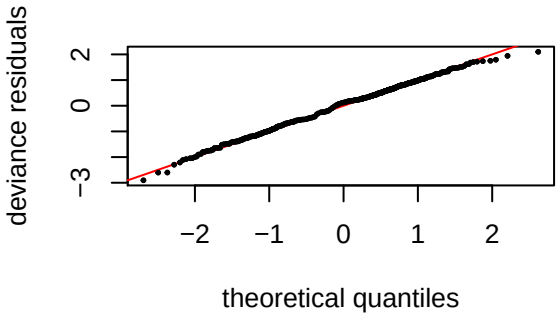

x0
x1
x2
x3
y/n
0.0
0.2
0.4
0.6
0.8
1.0
x0 + x1 + x2 + x3

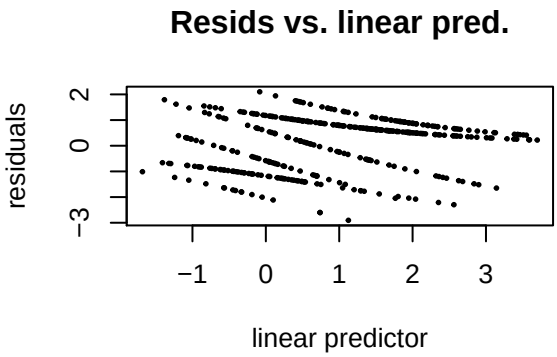

s(x0,1)
2
0
−2
0.0
0.2
0.4
0.6
0.8
1.0
x0

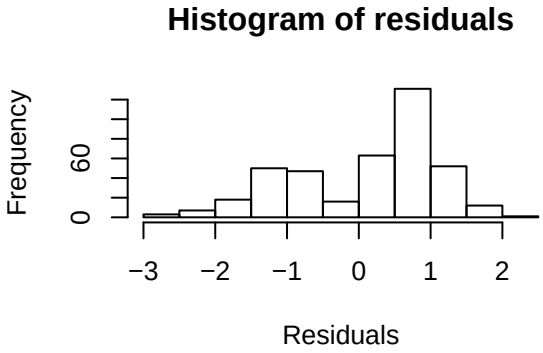

s(x1,2.31)
2
0
−2
0.0
0.2
0.4
0.6
0.8
1.0
x1

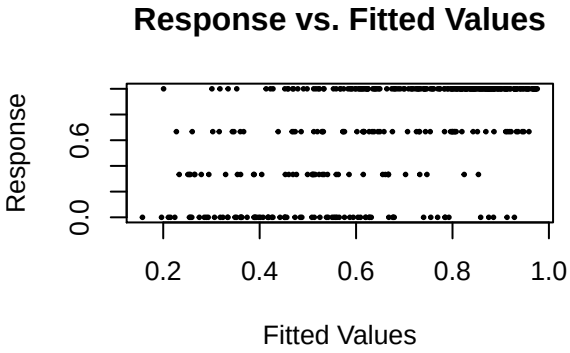

s(x2,6.02)
2
0
−2
0.0
0.2
0.4
0.6
0.8
1.0
x2


s(x3,1)
2
0
−2
0.0
0.2
0.4
0.6
0.8
1.0
x3

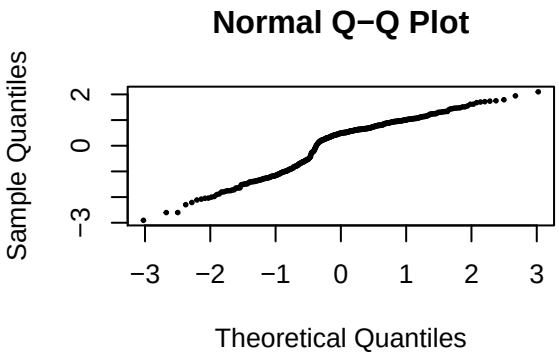


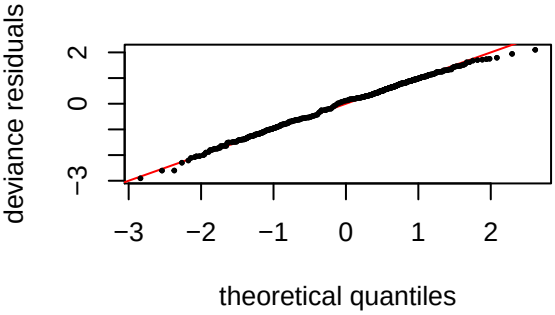


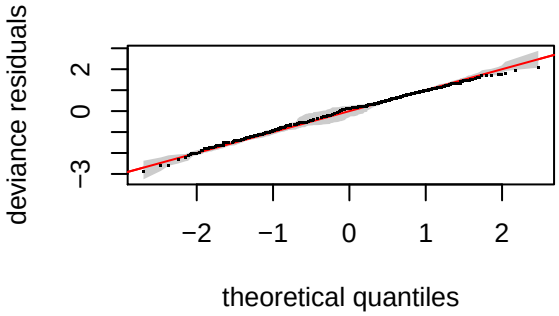


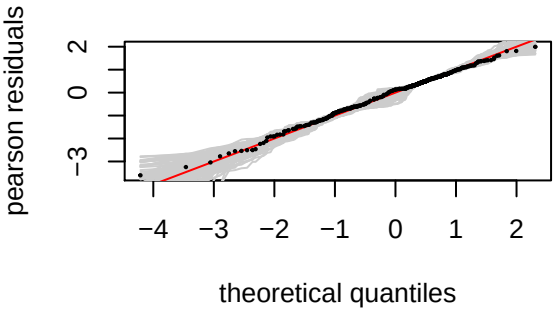



```
##
## Method: REML   Optimizer: outer newton
## full convergence after 9 iterations.
## Gradient range [-0.0001272,3.174e-06]
## (score 306.5 & scale 1).
## Hessian positive definite, eigenvalue range [4.285e-05,1.849].
## Model rank =  37 / 37
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##          k'   edf k-index p-value
## s(x0) 9.000 1.000   1.026    0.78
## s(x1) 9.000 2.314   0.905    0.06
## s(x2) 9.000 6.016   0.966    0.28
## s(x3) 9.000 1.000   1.020    0.68
```


Normal Q−Q Plot
Sample Quantiles
Theoretical Quantiles
−3
−2
−1
0
1
2
3


deviance residuals
theoretical quantiles
−3
−2
−1
0
1
2


deviance residuals
theoretical quantiles
−3
−2
−1
0
1
2


pearson residuals
theoretical quantiles
−4
−3
−2
−1
0
1
2