General Parametric Splines in carEx

2019-04-13

```
Cmat
```

```
function (knots, degree, smoothness, lin = NULL, intercept = 0,
    signif = 3)
{
    dm = max(degree)
    cmat = NULL
    if (!is.null(intercept))
        cmat = rbind(cmat, Intercept = Xf(intercept, knots, dm,
            D = 0))
    for (i in seq_along(knots)) {
        k <- knots[i]
        sm <- smoothness[[i]]
        if (max(sm) > -1) {
            if (!is.list(smoothness))
                sm <- 0:sm
            dmat <- Xf(k, knots, dm, D = sm, F) - Xf(k, knots,
                dm, D = sm, T)
            rownames(dmat) <- paste0("C", sm, "|", signif(k,
                signif))
            cmat <- rbind(cmat, dmat)
        }
    }
    degree <- rep(degree, length.out = length(knots) + 1)
    for (i in seq_along(degree)) {
        di <- degree[i]
        if (dm > di) {
            dmat <- diag((length(knots) + 1) * (dm + 1))[(i -
                1) * (dm + 1) + 1 + seq(di + 1, dm), , drop = F]
            rownames(dmat) = paste("I.", i, ".", seq(di + 1,
                dm), sep = "")
            cmat = rbind(cmat, dmat)
        }
    }
    if (!is.null(lin))
        cmat <- rbind(cmat, lin)
    rk = qr(cmat)$rank
    spline.rank = ncol(cmat) - rk
    attr(cmat, "ranks") = c(npar.full = ncol(cmat), C.n = nrow(cmat),
        C.rank = rk, spline.rank = spline.rank)
    attr(cmat, "d") = svd(cmat)$d
    cmat
}
<bytecode: 0x000000001aceb5c8>
<environment: 0x000000001ad17f08>
```

Introduction

The parametric polynomial splines implemented in the 'carEx' package are piecewise polynomial functions on $k+1$ intervals formed by k knots partitioning the real line:

$$(-\infty, t_1], (t_1, t_2], ..., (t_{i-1}, t_i], ..., (t_k, \infty)$$

with degree d_i on the ith interval $(t_{i-1}, t_i]$, $i = 1, ..., k+1$, and order of continuity c_i at the ith knot, $i = 1, ..., k$.

The order of continuity refers to the highest order for which the derivatives of the polynomial on the interval to the left and to the right of a knot, t_i, have the same limits at t_i. For all orders above c_i, derivatives, if any, are not constrained to have the same limit.

Such a spline is parametrized by three vectors: a vector of knots, $t_1 < t_2 < ... < t_k$, of length $k > 0$, a vector of polynomial degrees, $d_1, d_2, ..., d_{k+1}$, of length $k+1$, and a vector of orders of continuity or 'smoothness', $c_1, c_2, ..., c_k$, of length k.

Theory

We first describe the general principles that underly the implemention of splines in this package.

Let X_f be a $n \times q$ matrix for a model whose coefficients are subject to c linearly independent constraints given by a $c \times q$ matrix C. That is, the linear space for the model is:

$$\mathcal{M} = \{\eta = X_f \phi : \phi \in \mathbb{R}^q, C\phi = 0\}$$

We wish to construct a $n \times p$ design matrix X with $p = q - c$ so that

$$\mathcal{M} = \{\eta = X\beta : \beta \in \mathbb{R}^p\}$$

Suppose further that we want the parameters β to provide p specified linearly independent functions of ϕ represented by the rows of the $p \times q$ matrix E whose rows are linearly independent of the rows of C to ensure that they are not equal to 0 on $\mathcal{M}$.

Then the $q \times q$ partitioned matrix $\begin{bmatrix} C \\ E \end{bmatrix}$ has linearly independent rows and is invertible with a conformably partitioned inverse:

$$\begin{bmatrix} F & G \end{bmatrix} = \begin{bmatrix} C \\ E \end{bmatrix}^{-1}$$

Thus $FC + GE = I$, $CF = I$, etc.

Consider the model matrix $X = X_f G$. We show that $\mathcal{M} = \{X\beta : \beta \in \mathbb{R}^p\}$ and that for any $\phi \in \mathbb{R}^q$, such that $C\phi = 0$, $\beta = E\phi$.

Suppose $C\phi = 0$. Then

$$\phi = \begin{bmatrix} F & G \end{bmatrix} \begin{bmatrix} C \\ E \end{bmatrix} \phi = \begin{bmatrix} F & G \end{bmatrix} \begin{bmatrix} 0 \\ E\phi \end{bmatrix} = GE\phi$$

Thus, with $\beta = E\phi$, we have

$$X_f \phi = X_f GE\phi = X\beta$$

If X is of full rank, this defines a 1-1 correspondence between $\beta \in \mathbb{R}^p$ and $\{\phi \in \mathbb{R}^q : C\phi = 0\}$ given by $\beta = E\phi$ and $\phi = G\beta$.

We can obtain the least-squares estimator $\hat{\beta} = (X'X)^{-1}X'Y$. We can then estimate any linear function $\psi = L\phi$ of ϕ under the constraint $C\phi = 0$ with the estimator $\hat{\psi} = A\hat{\beta}$ with

$$A = LG$$

Thus, the matrix G serves as a post-multiplier to transform X_f into a model matrix $X = X_f G$ that can be used in a linear model.

The same matrix G also serves as a post-multiplier to transform any general linear hypothesis matrix expressed in terms of ϕ into a general linear hypothesis matrix in terms of of β.

Application to Splines

Our goal is to generate model matrices for splines in a way that produces interpretable coefficients and lends itself to easily estimating and testing properties of the spline that are linear functions of parameters: slope, curvature, discontinuities, etc.

Given k knots, $-\infty = t_0 < t_1 < ... < t_k < t_{k+1} = \infty$, the spline in the ith interval, $(t_{i-1}, t_i]$, is a polynomial of degree d_i, a non-negative integer with the value 0 signifying a constant over the corresponding interval.

The order of smoothness c_i at t_i is either a non-negative integer or -1 to allow a discontinuity.

Generating a model matrix for piecewise polynomial functions is sometimes simple. For example, if the degrees, d_i, are non-decreasing and the order of continuity is a constant c less than $\min(d_i)$, one can add terms using 'plus' functions at each knot. For example, a quadratic spline (degree 2, continuity 1) with one knot at 1 can be generated with a model matrix with three columns, in addition to the intercept term:

$$x, x^2, (x-1)_+^2$$

where

$$(y)_+ = \begin{cases} 0 & \text{if } y < 0 \\ y & \text{otherwise} \end{cases}$$

A spline that is quadratic on the interval $(-\infty, 1]$ and cubic on $(1, \infty)$ with continuity of order 1, $c_1 = 1$, at $t_1 = 1$, can be generated by the columns:

$$x, x^2, (x-1)_+^2, (x-1)_+^3$$

However, if one allows the degree of the polynomial or the order of smoothness to vary in different parts of the spline, the approach above works only in special cases.

Generating model matrices in more general situations, for example with degrees that are not monotone, nor monotone increasing as the index radiates from a central value, is more challenging. The approach described here works for any pattern of degrees, d_i and smoothness constraints, c_i.

We start by constucting a matrix, X_f, for a spline in which the polynomial degree in each interval is the maximal value, $\max(d_i)$. We then construct constraints for the coefficients of this model to produce the desired spline.

As an example, consider a spline, $\mathcal{S}$, with knots at 3 and 7, polynomial degrees, $(2, 3, 2)$, and smoothness, $(1, 2)$, meaning that $\mathcal{S}$ is smooth of order 1 at $x = 3$, and smooth of order 2 at $x = 7$. Columns of the full matrix X_f contain the intercept, linear and quadratic and cubic terms in each interval of the spline.

To create an instance of X_f we need to specify the values over which the matrix is evaluated. Evaluating X_f at $x = 0, 1, ...9$, we obtain the following matrix, which happens here to be block diagonal because of the ordering of the x values:

```
Xf(0:9, knots = c(3,7), degree = 3)
```

```
     X0 X1 X2 X3 X0 X1 X2  X3 X0 X1 X2  X3
f(0)  1  0  0  0  0  0  0   0  0  0  0   0
f(1)  1  1  1  1  0  0  0   0  0  0  0   0
f(2)  1  2  4  8  0  0  0   0  0  0  0   0
f(3)  1  3  9 27  0  0  0   0  0  0  0   0
f(4)  0  0  0  0  1  4 16  64  0  0  0   0
f(5)  0  0  0  0  1  5 25 125  0  0  0   0
f(6)  0  0  0  0  1  6 36 216  0  0  0   0
f(7)  0  0  0  0  1  7 49 343  0  0  0   0
f(8)  0  0  0  0  0  0  0   0  1  8 64 512
f(9)  0  0  0  0  0  0  0   0  1  9 81 729
attr(,"class")
[1] "gspline_matrix" "matrix"
```

The model for the unconstrained maximal polynomial is $X_f \phi : \phi \in \mathbb{R}^{12}$.

We impose three types of constraints on ϕ.

1. $X_f \phi$ should evaluate to 0 at $x = 0$ so an intercept term in the model will have the correct interpretation,
2. the limits of the value and of the first derivative of the spline must be the same when approaching the first knot from the right or from the left, and the limits of the value, the first and second derivatives should be the same when approaching the second knot from the right or from the left, and
3. the degree of the polynomial in the first and third intervals must be reduced to 2.

The constraint marix, C is created by the 'Cmat' function:

```
Cmat(knots = c(3, 7), degree = c(2, 3, 2), smooth = c(1, 2))
```

```
      X0 X1 X2  X3 X0 X1  X2   X3 X0 X1 X2  X3
f(0)   1  0  0   0  0  0   0    0  0  0  0   0
C0|3  -1 -3 -9 -27  1  3   9   27  0  0  0   0
C1|3   0 -1 -6 -27  0  1   6   27  0  0  0   0
C0|7   0  0  0   0 -1 -7 -49 -343  1  7 49 343
C1|7   0  0  0   0  0 -1 -14 -147  0  1 14 147
C2|7   0  0  0   0  0  0  -2  -42  0  0  2  42
I.1.3  0  0  0   1  0  0   0    0  0  0  0   0
I.3.3  0  0  0   0  0  0   0    0  0  0  0   1
attr(,"ranks")
  npar.full         C.n      C.rank spline.rank
         12           8           8           4
attr(,"d")
[1] 536.66701452  48.80391245  10.85308819   3.18591258   0.97504352
[6]   0.81688866   0.35905212   0.08458296
```

The row labels of the constraint matrix show the role of each row. For example, "f(0)" is the value of the spline when $x = 0$ which is constrained to 0 so that an intercept term in a linear model can have its usual interpretation, "C(3).0" ensures continuity at $x = 3$, "C(7).2" forces continuity of the second derivative at $x = 7$, "I.1.3" constrains the cubic term to be 0 in the first interval, etc.

The 'd' attribute contains the vector of singular values of the constraint matrix.

The following is the matrix E of estimable functions created by the 'Emat' function:

```
Emat(knots = c(3, 7), degree = c(2, 3, 2), smooth = c(1, 2))
```

```
     X0 X1 X2  X3 X0 X1 X2 X3 X0 X1 X2 X3
D1|0  0  1  0   0  0  0  0  0  0  0  0  0
D2|0  0  0  2   0  0  0  0  0  0  0  0  0
C2|3  0  0 -2 -18  0  0  2 18  0  0  0  0
```

```
C3|3  0  0  0  -6  0  0  0  6  0  0  0  0
```

The row labels signify the first derivative at $x = 0$, 'D1(0)', the second derivative at $x = 0$, 'D2(0)', the saltus in the second derivative at $x = 3$ and the saltus in the third derivative at $x = 3$.

The full rank model for the spline is generated by a matrix $X = X_f G$ as described in the previous section.

The spline modelling function is a closure generated by the `gspline` function.

```
sp <- gspline(knots = c(3, 7), degree = c(2, 3, 2), smoothness = c(1, 2))
sp(0:9)
```

```
     D1|0 D2|0 C2|3     C3|3
f(0)    0  0.0  0.0  0.00000
f(1)    1  0.5  0.0  0.00000
f(2)    2  2.0  0.0  0.00000
f(3)    3  4.5  0.0  0.00000
f(4)    4  8.0  0.5  0.16667
f(5)    5 12.5  2.0  1.33333
f(6)    6 18.0  4.5  4.50000
f(7)    7 24.5  8.0 10.66667
f(8)    8 32.0 12.5 20.66667
f(9)    9 40.5 18.0 34.66667
attr(,"class")
[1] "gspline_matrix" "matrix"
```

produce a matrix $X = X_f G$ that will generate the desired spline parametrized by linear estimable coefficients.

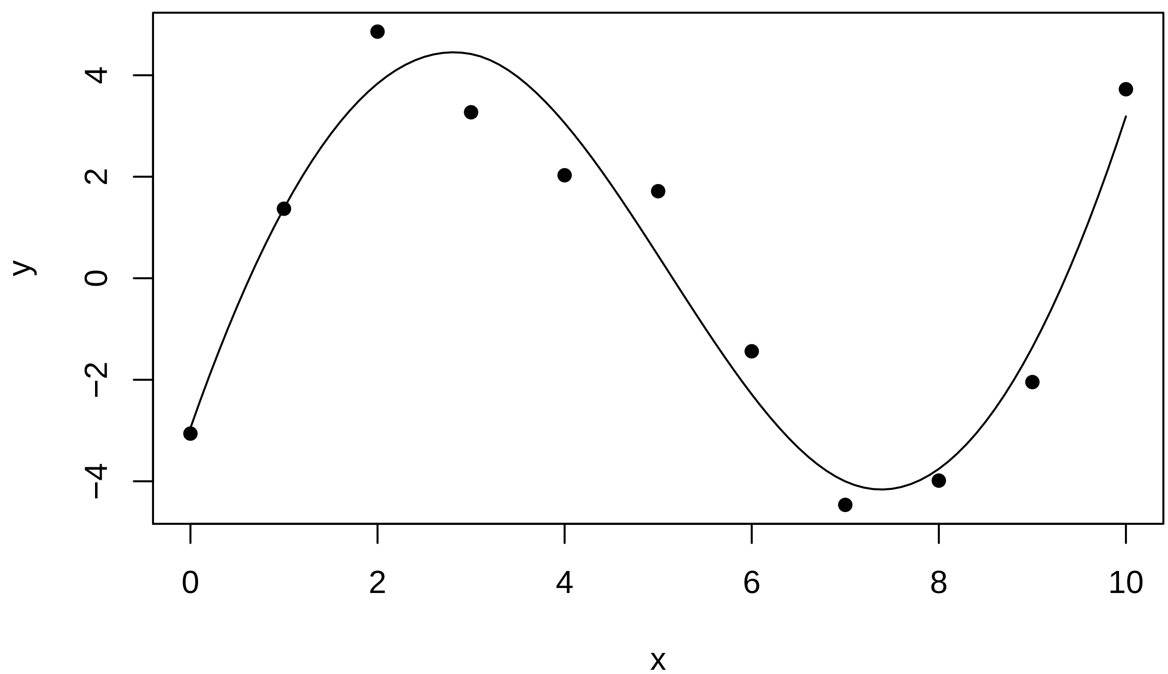
The closure created by the `gspline` function can be used in a linear model formulas. We illustrate its use with a small example. Note that the spline function can be used in any linear model formula. It can, for example, be modelled as interacting with other predictors.

```
df <- data.frame(x = 0:10)
set.seed(123)
df <- within(df, y <- -2* (x-5) + .1 * (x-5)^3 + rnorm(x))
df <- rbind(df, data.frame(x = seq(0,10,.1), y = NA))
df <- sortdf(df, ~ x)
plot(y~x, df, pch = 16)
fit <- lm(y ~ sp(x), data = df)
summary(fit)
```

```

Call:
lm(formula = y ~ sp(x), data = df)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.1476 -0.5748 -0.1091  0.6914  1.2704 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)   
(Intercept)  -2.9513     1.0165  -2.903  0.02721 * 
sp(x)D1|0     5.2685     1.3117   4.017  0.00699 **
sp(x)D2|0    -1.8747     0.6726  -2.787  0.03169 * 
sp(x)C2|3    -0.5129     1.3846  -0.370  0.72381   
sp(x)C3|3     1.1346     0.2749   4.127  0.00616 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 1.064 on 6 degrees of freedom
  (101 observations deleted due to missingness)
Multiple R-squared:  0.9372,    Adjusted R-squared:  0.8954
F-statistic:  22.4 on 4 and 6 DF,  p-value: 0.0009419
```

```r
lines(df$x , predict(fit, df))
```



Linear hypotheses

Linear hypotheses about a spline may be easy to formulate in terms of its 'full' parameter vector ϕ but challenging in terms of the 'working' parameters, β. For example, the derivative or curvature of the spline over a range of values is easily expressed in terms of ϕ. To do this We use the relationship between linear hypotheses in terms of ϕ with those in terms of β to generate linear hypotheses based on $\hat{\beta}$. Namely the least-squares estimator of $\psi = L\phi$ under the contraint $C\phi = 0$ is $\hat{\psi} = A\hat{\beta}$ where $A = LG$.

Given a spline function `sp` created by the `gspline` function:

```r
sp <- gspline(knots = c(3,7), degree = c(2,3,2), smoothness = c(1,2))
sp(0:9)
```

```
     D1|0 D2|0 C2|3     C3|3
f(0)    0  0.0  0.0  0.00000
f(1)    1  0.5  0.0  0.00000
f(2)    2  2.0  0.0  0.00000
f(3)    3  4.5  0.0  0.00000
f(4)    4  8.0  0.5  0.16667
```

```
f(5)    5 12.5  2.0  1.33333
f(6)    6 18.0  4.5  4.50000
f(7)    7 24.5  8.0 10.66667
f(8)    8 32.0 12.5 20.66667
f(9)    9 40.5 18.0 34.66667
attr(,"class")
[1] "gspline_matrix" "matrix"
```

The `sp` function will generate a hypothesis matrix to query values and derivatives of the spline.

```
sp(c(2, 3, 7), D = 1)
```

```
     D1|0 D2|0 C2|3 C3|3
D1|2    1    2    0    0
D1|3    1    3    0    0
D1|7    1    7    4    8
attr(,"class")
[1] "gspline_matrix" "matrix"
```

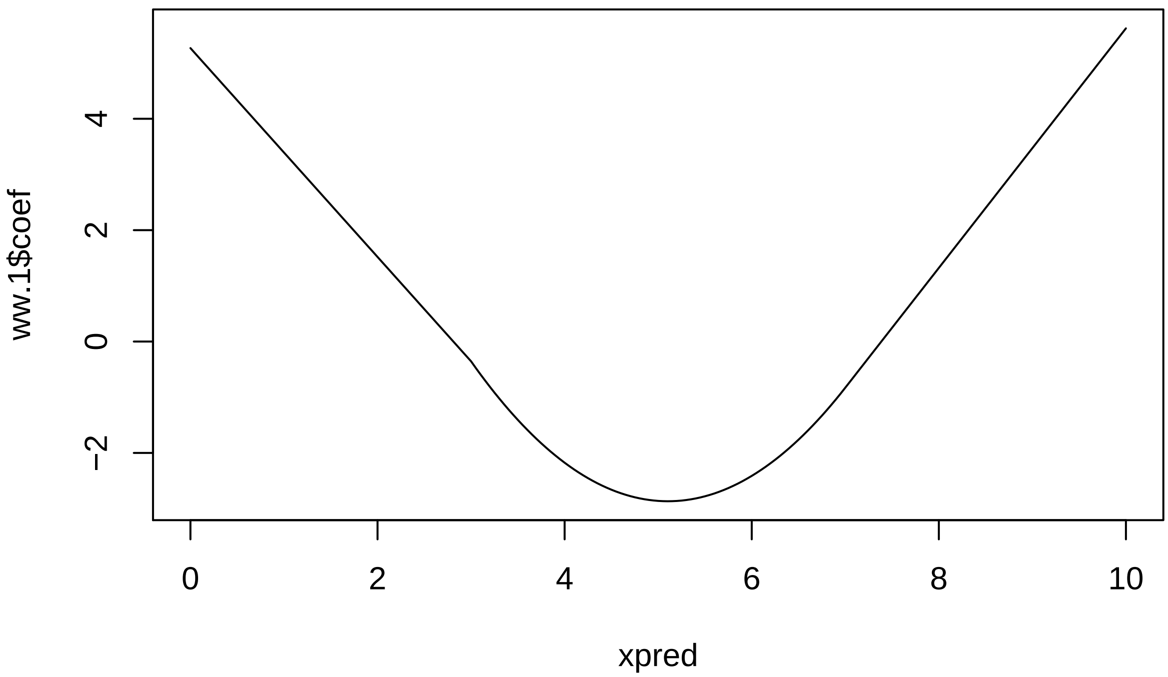
Denoting the matrix above by A, $A\hat{\beta}$ will estimate the first derivative of the spline at $x = 2$ and its limit from the right at the knots $x = 3, 7$. The `limit` parameter to the spline function is used to select whether the value estimated is a limit from the right, from the left, or the saltus (jump) in value if discontinuous. For example, at $x = 3$ where the spline has a discontinuous second derivatives:

```
sp(c(3, 3, 3), D = 2, limit = c(-1,0,1))
```

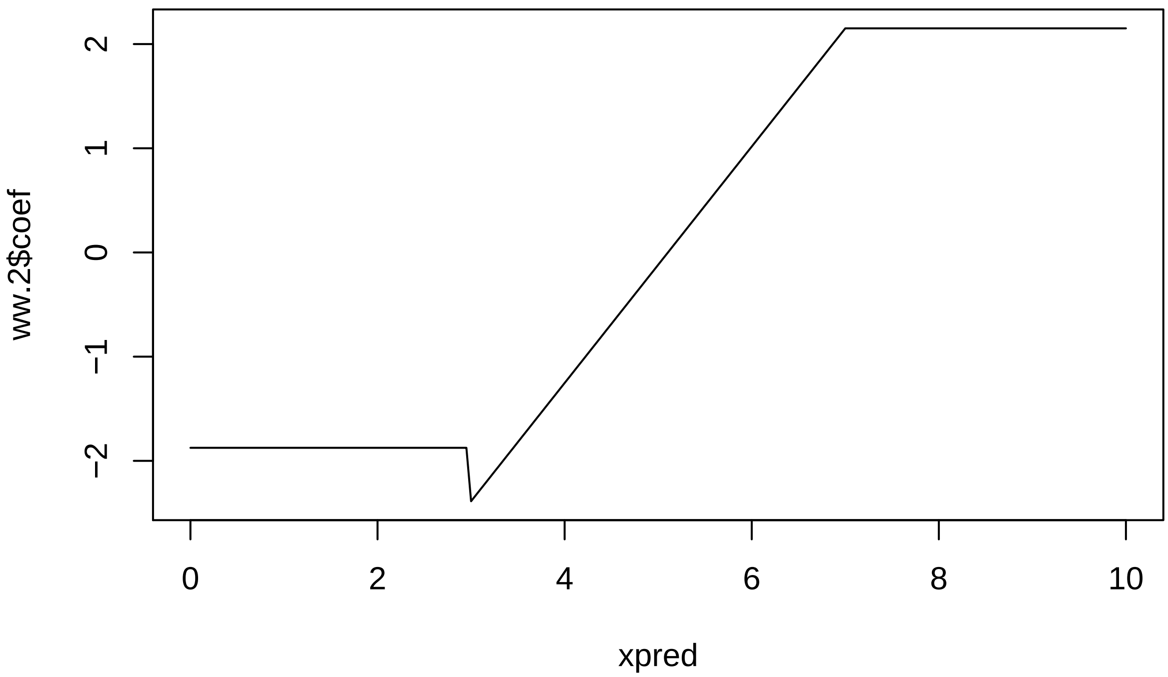
```
            D1|0 D2|0 C2|3 C3|3
D2|3-          0    1    0    0
D2|3+-D2|3-    0    0    1    0
D2|3+          0    1    1    0
attr(,"class")
[1] "gspline_matrix" "matrix"
```

Using the 'wald' function it is possible to graph these estimates as a function of of x.

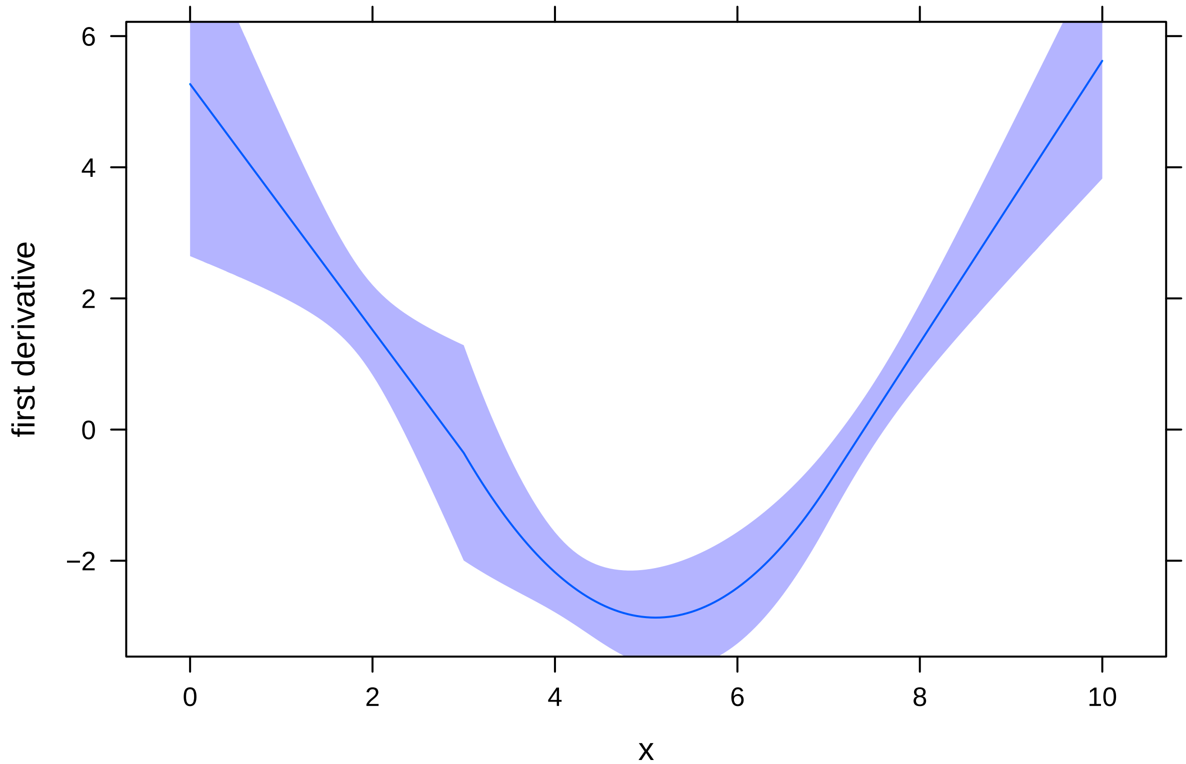
```
xpred <- seq(0,10, .05)

A.1 <- cbind(0, sp(xpred, D = 1))
ww.1 <- as.data.frame(wald(fit, A.1))

A.2 <- cbind(0, sp(xpred, D = 2))
ww.2 <- as.data.frame(wald(fit, A.2))

plot(xpred, ww.1$coef, type = 'l')
```




```
plot(xpred, ww.2$coef, type = 'l')
```




```r
library(latticeExtra)
ww.1$x <- xpred
xyplot(coef ~ x, ww.1, type = 'l',
       lower = ww.1$L2, upper = ww.1$U2,
       ylab = 'first derivative',
       subscripts = TRUE) +
    layer(gpanel.fit(...))
```




```r
head(ww.1)
```

```
             coef       se       U2       L2    x
D1|0    5.268497 1.311704 7.891905 2.645089 0.00
D1|0.05 5.174763 1.279216 7.733194 2.616331 0.05
D1|0.1  5.081028 1.246788 7.574604 2.587453 0.10
D1|0.15 4.987294 1.214425 7.416144 2.558444 0.15
D1|0.2  4.893560 1.182133 7.257826 2.529293 0.20
D1|0.25 4.799825 1.149918 7.099661 2.499989 0.25
```

Discontinuity

We use the monthly U.S. unemployment rates from January 1995 to February 2019 to illustrate a model with a discontinuity and, subsequently, we will show a periodic spline component can be added to model periodic patterns such as annual seasonal patterns.

The 'crash' in November 2008 creates a discontinuity in the series which we will treat as an 'a priori' discontinuity.

```r
unemp <- read.csv('http://blackwell.math.yorku.ca/data/USUnemployment.csv')
unemp$date <- as.Date(unemp$date)
head(unemp)
```

```
        date unemployment
1 1995-01-01         10.6
2 1995-02-01         10.4
```

```
3 1995-03-01          10.7
4 1995-04-01          10.0
5 1995-05-01           9.6
6 1995-06-01           9.2
```

```
library(lattice)
library(latticeExtra)
xyplot(unemployment ~ date, unemp) + layer(panel.abline(v = as.Date('2008-12-15', col = 'blue')))
```


```
toyear <- function(x) {
  # number of years from January 1, 2000
  (as.numeric(x) - as.numeric(as.Date('2000-01-01')))/365.25
}
unemp <- within(
  unemp,
  {
    year <- toyear(date)
    month <- as.numeric(format(date, '%m'))
  })
summary(unemp)
```

```
      date             unemployment        month            year
 Min.   :1995-01-01   Min.   : 5.200   Min.   : 1.000   Min.   :-4.999
 1st Qu.:2001-01-08   1st Qu.: 6.600   1st Qu.: 3.000   1st Qu.: 1.023
 Median :2007-01-16   Median : 7.300   Median : 6.000   Median : 7.043
 Mean   :2007-01-15   Mean   : 7.448   Mean   : 6.466   Mean   : 7.041
 3rd Qu.:2013-01-24   3rd Qu.: 8.100   3rd Qu.: 9.000   3rd Qu.:13.066
```

```
 Max.   :2019-02-01   Max.   :10.700   Max.   :12.000   Max.   :19.086
```

The following code creates a quadratic spline and a cubic spline with knots at quintiles.

```
quintiles <- quantile(unemp$year, 1:4/5)
sp2 <- gspline(quintiles, 2, 1) # quadratic spline
sp3 <- gspline(quintiles, 3, 2) # cubic spline
```

We can also add a knot at the point of discontinuity coincident with the 2008 crash.

```
quintiles_with_crash <- sort(c(quintiles, toyear(as.Date('2008-12-15'))))

sp2d <- gspline(quintiles_with_crash, 2, c(1,1,-1,1,1))
sp3d <- gspline(quintiles_with_crash, 3,  c(2,2,-1,2,2))
```

The following code fits four models using a quadratic or cubic spline with or without a discontinuity.

```
fit2 <- lm(unemployment ~ sp2(year), unemp)
summary(fit2)
```

```

Call:
lm(formula = unemployment ~ sp2(year), data = unemp)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.51604 -0.49821 -0.01906  0.47250  1.90185 

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)    
(Intercept)         7.82474    0.09008  86.868  < 2e-16 ***
sp2(year)D1|0      -0.17468    0.06700  -2.607  0.00961 ** 
sp2(year)D2|0      -0.01409    0.02529  -0.557  0.57785    
sp2(year)C2|-0.184 -0.17031    0.06748  -2.524  0.01216 *  
sp2(year)C2|4.63    0.14677    0.04567   3.214  0.00146 ** 
sp2(year)C2|9.45   -0.28896    0.04569  -6.324  9.9e-10 ***
sp2(year)C2|14.3    0.17277    0.06750   2.560  0.01100 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7104 on 283 degrees of freedom
Multiple R-squared:  0.6344,    Adjusted R-squared:  0.6266 
F-statistic: 81.84 on 6 and 283 DF,  p-value: < 2.2e-16
```

```
unemp$fit2 <- predict(fit2)
fit3 <- lm(unemployment ~ sp3(year), unemp)
summary(fit3)
```

```

Call:
lm(formula = unemployment ~ sp3(year), data = unemp)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.65225 -0.53935  0.01539  0.51200  1.94448 

Coefficients:
```

```
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)          7.728274   0.115711  66.789  < 2e-16 ***
sp3(year)D1|0       -0.376882   0.047831  -7.879 7.19e-14 ***
sp3(year)D2|0        0.104451   0.055240   1.891   0.0597 .
sp3(year)D3|0       -0.011672   0.019865  -0.588   0.5573
sp3(year)C3|-0.184 -0.057021   0.071984  -0.792   0.4289
sp3(year)C3|4.63     0.002714   0.034758   0.078   0.9378
sp3(year)C3|9.45    -0.018936   0.034765  -0.545   0.5864
sp3(year)C3|14.3     0.058433   0.071957   0.812   0.4174
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7385 on 282 degrees of freedom
Multiple R-squared:  0.6062,	Adjusted R-squared:  0.5965
F-statistic: 62.02 on 7 and 282 DF,  p-value: < 2.2e-16
```

```r
unemp$fit3 <- predict(fit3)

fit2d <- lm(unemployment ~ sp2d(year), unemp)
summary(fit2d)
```

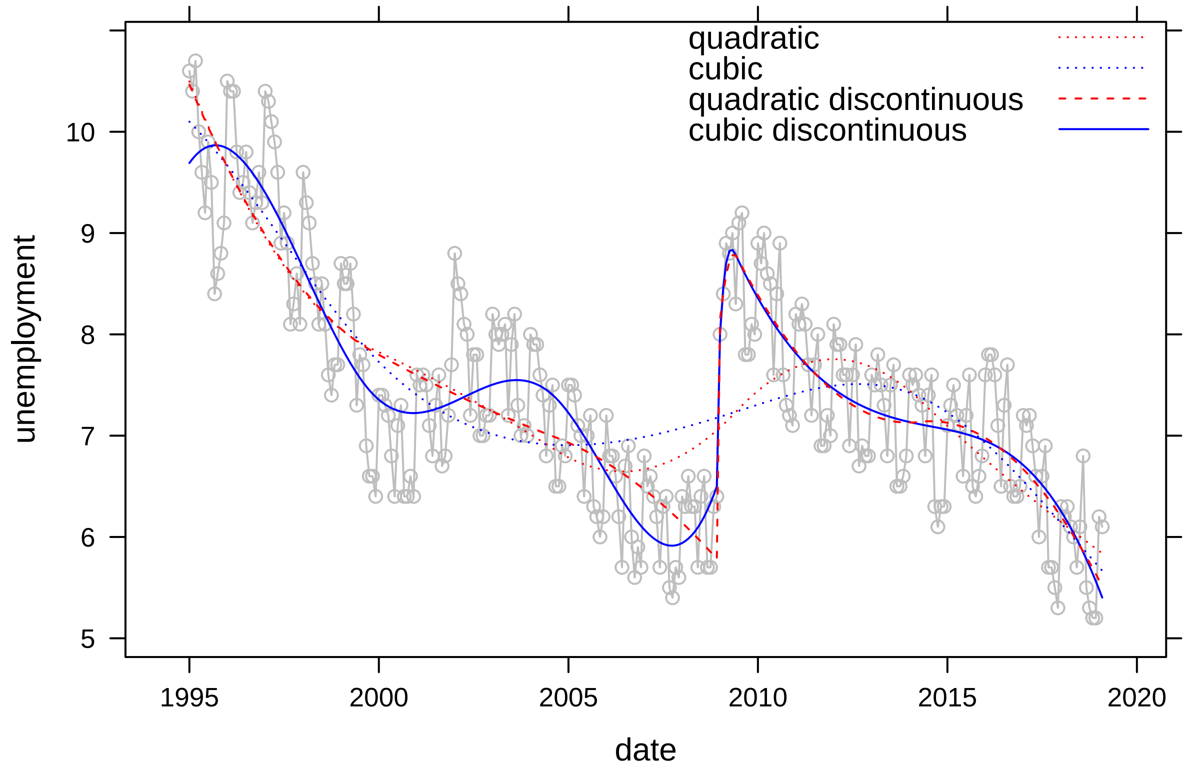
```
Call:
lm(formula = unemployment ~ sp2d(year), data = unemp)

Residuals:
     Min       1Q   Median       3Q      Max
-1.50968 -0.44994  0.05091  0.45359  1.41990

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          7.79826    0.07585 102.815  < 2e-16 ***
sp2d(year)D1|0      -0.20751    0.05896  -3.520 0.000504 ***
sp2d(year)D2|0       0.01405    0.02526   0.556 0.578491
sp2d(year)C2|-0.184 -0.12543    0.06087  -2.061 0.040267 *
sp2d(year)C2|4.63   -0.07886    0.07068  -1.116 0.265540
sp2d(year)C0|8.96    2.21150    0.61367   3.604 0.000371 ***
sp2d(year)C1|8.96    4.32041    2.67733   1.614 0.107719
sp2d(year)C2|8.96   -9.30985    5.55410  -1.676 0.094813 .
sp2d(year)C2|9.45    9.53430    5.57228   1.711 0.088184 .
sp2d(year)C2|14.3   -0.31857    0.07899  -4.033 7.11e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5864 on 280 degrees of freedom
Multiple R-squared:  0.7535,	Adjusted R-squared:  0.7455
F-statistic: 95.08 on 9 and 280 DF,  p-value: < 2.2e-16
```

```r
unemp$fit2d <- predict(fit2d)
fit3d <- lm(unemployment ~ sp3d(year), unemp)
summary(fit3d)
```

```
Call:
lm(formula = unemployment ~ sp3d(year), data = unemp)
```

```
Residuals:
     Min       1Q   Median       3Q      Max 
-1.46661 -0.43416  0.04281  0.39201  1.46187 

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)    
(Intercept)           7.35489    0.08890  82.736  < 2e-16 ***
sp3d(year)D1|0       -0.31782    0.03501  -9.079  < 2e-16 ***
sp3d(year)D2|0        0.43791    0.04935   8.874  < 2e-16 ***
sp3d(year)D3|0       -0.19288    0.02219  -8.692 3.13e-16 ***
sp3d(year)C3|-0.184  -0.46972    0.06367  -7.377 1.87e-12 ***
sp3d(year)C3|4.63     0.56217    0.07847   7.164 7.03e-12 ***
sp3d(year)C0|8.96     1.12727    0.68856   1.637   0.1027    
sp3d(year)C1|8.96     7.17553    4.63633   1.548   0.1228    
sp3d(year)C2|8.96   -38.30263   19.38613  -1.976   0.0492 *  
sp3d(year)C3|8.96    75.58355   39.50921   1.913   0.0568 .  
sp3d(year)C3|9.45   -76.00194   39.53243  -1.923   0.0556 .  
sp3d(year)C3|14.3    -0.04238    0.10074  -0.421   0.6743    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5308 on 278 degrees of freedom
Multiple R-squared:  0.7995,    Adjusted R-squared:  0.7916 
F-statistic: 100.8 on 11 and 278 DF,  p-value: < 2.2e-16
```

```r
unemp$fit3d <- predict(fit3d)

pp <- xyplot(unemployment ~ date, unemp, type = 'b',
             col = 'gray',
             key = list(
               corner = c(1,1),
               text = list(lab = c('quadratic','cubic','quadratic discontinuous','cubic discontinuous'))
               lines = list(col= c('red','blue','red','blue'),
                            lty = c(3,3,2,1))
               )) +
  layer(panel.lines(x, unemp$fit3, col = 'blue', lty = 3)) +
  layer(panel.lines(x, unemp$fit2, col = 'red', lty = 3)) +
  layer(panel.lines(x, unemp$fit3d, col = 'blue', lty = 1)) +
  layer(panel.lines(x, unemp$fit2d, col = 'red', lty = 2))
pp
```



The cubic model follows the data better but overestimates in the vicinity of 2000 and 2004. It also misses an upturn in 2016. The following is comparison of AICs for the four models.

```
AIC(fit2, fit3, fit2d, fit3d)
```

```
      df      AIC
fit2   8 633.5505
fit3   9 657.0742
fit2d 11 525.2724
fit3d 13 469.3494
```

Periodic spline

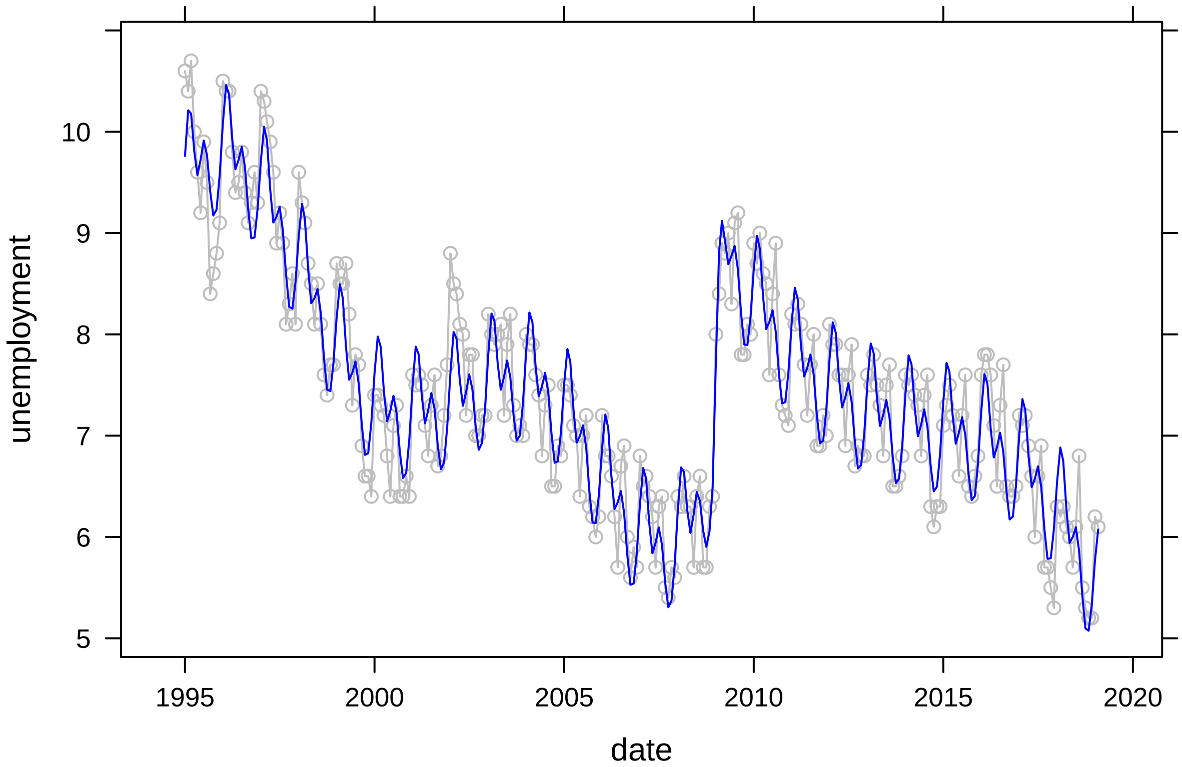
We add a periodic spline component as a function of months using a cubic spline with period 12 and four internal knot at months $12 \times (1/52/53/54/5)$.

```
per3 <- gspline(12 * 1:5/5, 3, 2, periodic = TRUE)
fitper3 <-  lm(unemployment ~ sp3d(year) + per3(month), unemp)
summary(fitper3)
```

```

Call:
lm(formula = unemployment ~ sp3d(year) + per3(month), data = unemp)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.01225 -0.23279 -0.00892  0.20521  1.18149 
```

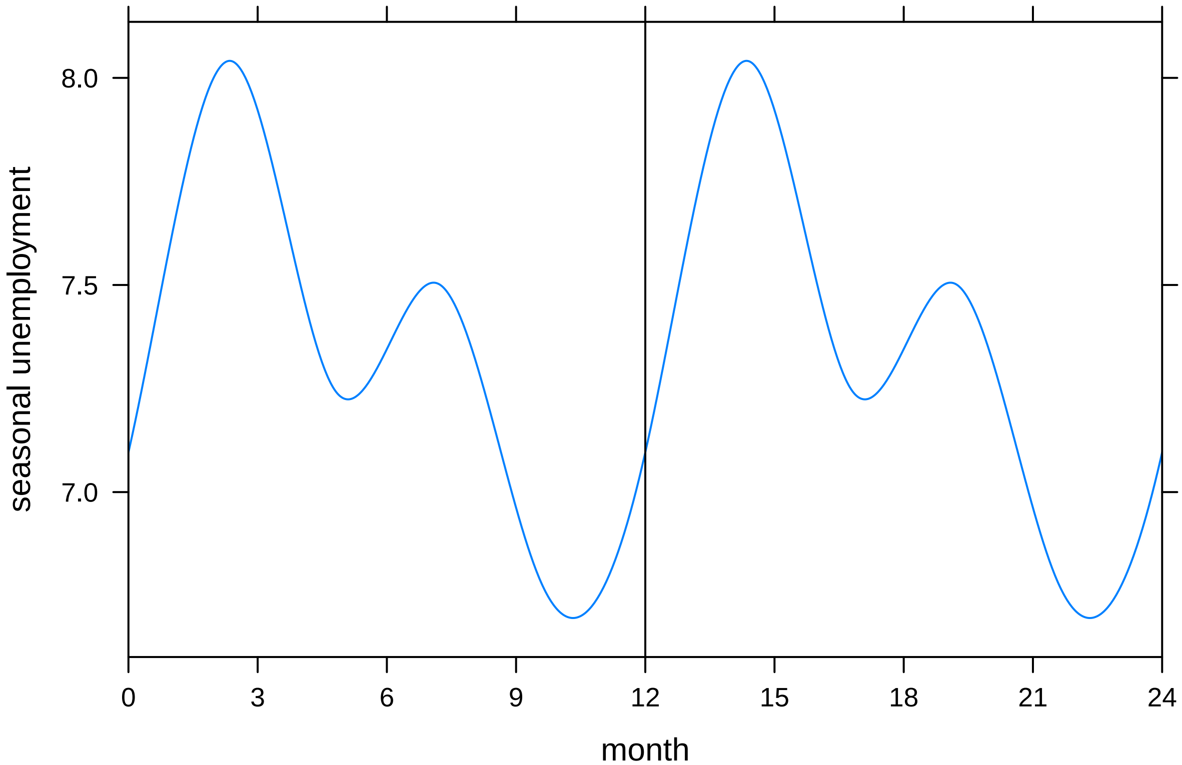
```
Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)            7.09603    0.07537  94.143  < 2e-16 ***
sp3d(year)D1|0        -0.32959    0.02339 -14.091  < 2e-16 ***
sp3d(year)D2|0         0.46948    0.03300  14.225  < 2e-16 ***
sp3d(year)D3|0        -0.20769    0.01484 -13.994  < 2e-16 ***
sp3d(year)C3|-0.184   -0.52094    0.04262 -12.223  < 2e-16 ***
sp3d(year)C3|4.63      0.62740    0.05253  11.942  < 2e-16 ***
sp3d(year)C0|8.96      0.11441    0.46534   0.246 0.805973    
sp3d(year)C1|8.96     11.37552    3.12632   3.639 0.000327 ***
sp3d(year)C2|8.96    -56.50467   13.07145  -4.323 2.16e-05 ***
sp3d(year)C3|8.96    112.04499   26.63734   4.206 3.52e-05 ***
sp3d(year)C3|9.45   -112.50572   26.65288  -4.221 3.31e-05 ***
sp3d(year)C3|14.3     -0.05644    0.06728  -0.839 0.402317    
per3(month)D1|12/12    0.45981    0.02659  17.290  < 2e-16 ***
per3(month)D2|12/12    0.24369    0.05616   4.339 2.02e-05 ***
per3(month)D3|12-/12  -0.03609    0.04414  -0.818 0.414265    
per3(month)C3|2.4/2.4  0.87800    0.08373  10.486  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3545 on 274 degrees of freedom
Multiple R-squared:  0.9119,    Adjusted R-squared:  0.907 
F-statistic:   189 on 15 and 274 DF,  p-value: < 2.2e-16
```

```r
unemp$fitper3 <- predict(fitper3)
pp <- xyplot(unemployment ~ date, unemp, type = 'b',
             col = 'gray') +
  layer(panel.lines(x, unemp$fitper3, col = 'blue'))
pp
```



We can examine the monthly periodic spline fit

```
pred <- data.frame(month = seq(0,24,.01), year = 0)
pred$fitper3 <- predict(fitper3, newdata = pred)
xyplot(fitper3 ~ month, pred, type = 'l',
       xlim = c(0,24),
       scales = list( x = list( at =3 * 0:8)),
       ylab = 'seasonal unemployment') +
  layer(panel.abline(v = 12))
```

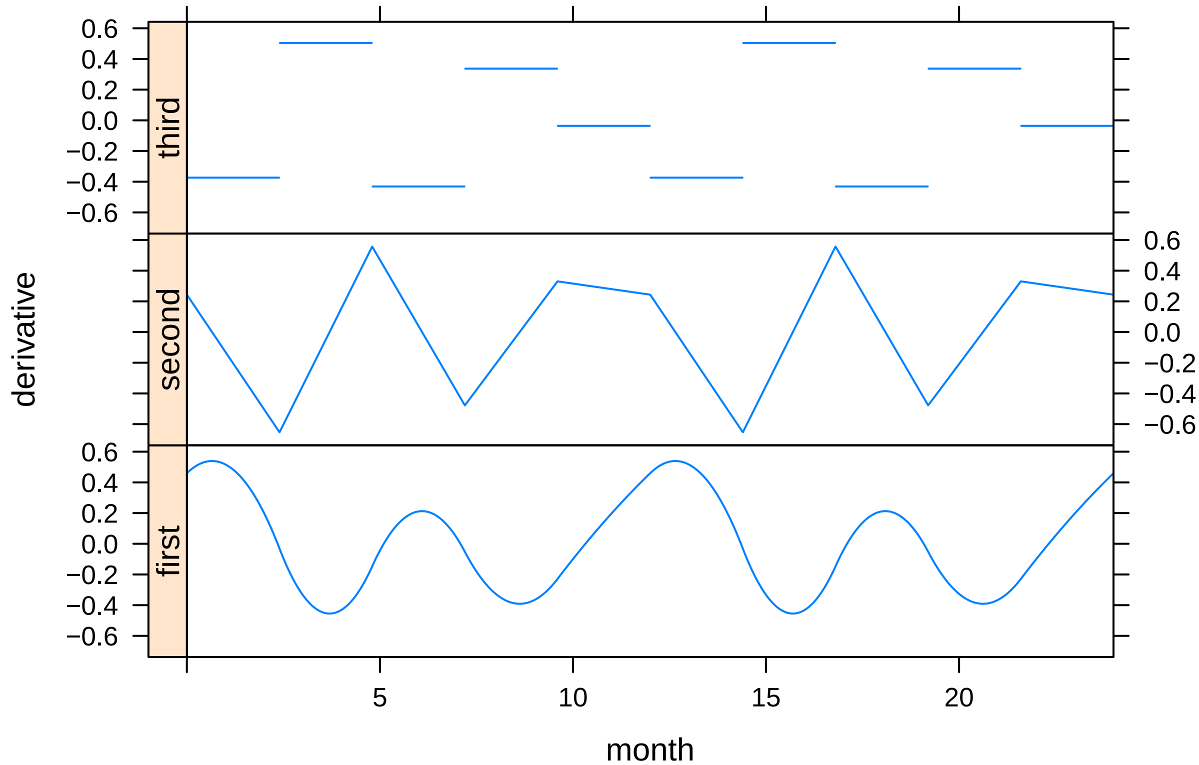


Although, the cubic periodic spline has only three free parameters fully determined by the first three derivatives on the left of 0, the third derivative is not continuous as the following plot of derivatives shows.

```
derivs <- expand.grid(month = seq(0, 24, .01), D = 1:3)
Ld <- with(derivs, per3(month, D = D, limit = -1))
Ld <- cbind(0*Ld[,rep(1,12)], Ld)
derivs <- cbind(derivs, walddf(fitper3, Ld))
```

```
Warning in wald(fit = fit, Llist = Llist, clevel = clevel, data = data, :
Poorly conditioned L matrix, calculated numDF may be incorrect
```

```
derivs $ order <-
  factor(c('first','second','third')[derivs$D])
derivs $ order <- with(derivs, reorder(order, D))
inds <- which(derivs$month %% (12/5) < .0001 & derivs$D == 3)
derivs$coef[inds] <- NA
xyplot(coef ~ month | order , derivs, type = 'l', layout = c(1,3),
       xlim = c(0,24),
       ylab = 'derivative',
       strip.left = T, strip = F)
```




```r
Ldi <- per3(seq(0,24,12/10), D = 3)
```

Now, using `Lfx`:

```r
summary(fitper3)
```

```

Call:
lm(formula = unemployment ~ sp3d(year) + per3(month), data = unemp)

Residuals:
     Min       1Q   Median       3Q      Max
-1.01225 -0.23279 -0.00892  0.20521  1.18149

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)              7.09603    0.07537  94.143  < 2e-16 ***
sp3d(year)D1|0          -0.32959    0.02339 -14.091  < 2e-16 ***
sp3d(year)D2|0           0.46948    0.03300  14.225  < 2e-16 ***
sp3d(year)D3|0          -0.20769    0.01484 -13.994  < 2e-16 ***
sp3d(year)C3|-0.184     -0.52094    0.04262 -12.223  < 2e-16 ***
sp3d(year)C3|4.63        0.62740    0.05253  11.942  < 2e-16 ***
sp3d(year)C0|8.96        0.11441    0.46534   0.246 0.805973
sp3d(year)C1|8.96       11.37552    3.12632   3.639 0.000327 ***
sp3d(year)C2|8.96      -56.50467   13.07145  -4.323 2.16e-05 ***
sp3d(year)C3|8.96      112.04499   26.63734   4.206 3.52e-05 ***
sp3d(year)C3|9.45     -112.50572   26.65288  -4.221 3.31e-05 ***
```

```
sp3d(year)C3|14.3       -0.05644    0.06728  -0.839 0.402317
per3(month)D1|12/12      0.45981    0.02659  17.290  < 2e-16 ***
per3(month)D2|12/12      0.24369    0.05616   4.339 2.02e-05 ***
per3(month)D3|12-/12    -0.03609    0.04414  -0.818 0.414265
per3(month)C3|2.4/2.4    0.87800    0.08373  10.486  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3545 on 274 degrees of freedom
Multiple R-squared:  0.9119,    Adjusted R-squared:  0.907
F-statistic:   189 on 15 and 274 DF,  p-value: < 2.2e-16
```

```r
Lfx(fitper3)
```

```
list( 1,
1 * M(sp3d(year)),
1 * M(per3(month))
)
```

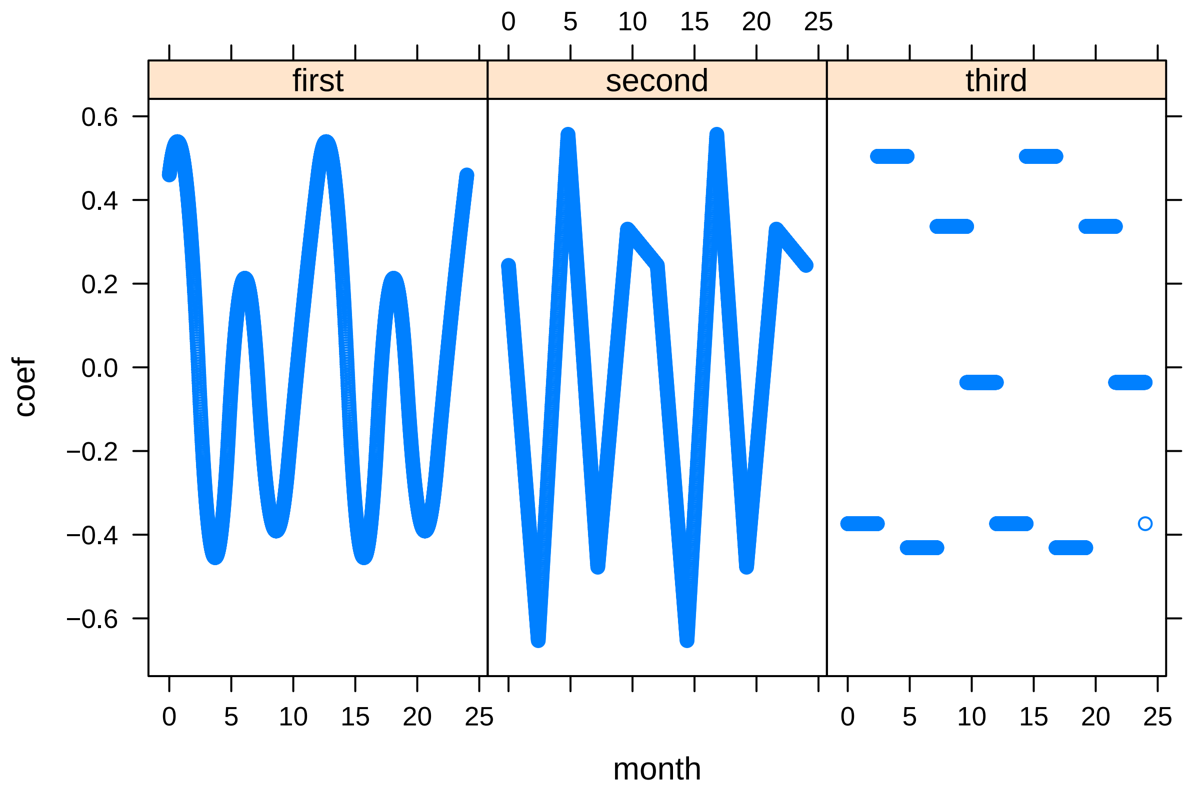
```r
derivs$year <- 0
ww <- walddf(
  fitper3,
  Lfx(fitper3,
      list( 0,
0 * M(sp3d(year)),
1 * M(per3(month, D = D))),
data = derivs))
```

```
Warning in wald(fit = fit, Llist = Llist, clevel = clevel, data = data, :
Poorly conditioned L matrix, calculated numDF may be incorrect
```

```r
head(ww)
```

```
                coef         se        U2        L2 month D      coef
D1.12.12   0.4598073 0.02659434 0.5129960 0.4066186  0.00 1 0.4598073
D1.0.01.12 0.4622255 0.02660005 0.5154256 0.4090254  0.01 1 0.4622255
D1.0.02.12 0.4646064 0.02661603 0.5178385 0.4113743  0.02 1 0.4646064
D1.0.03.12 0.4669499 0.02664200 0.5202339 0.4136659  0.03 1 0.4669499
D1.0.04.12 0.4692560 0.02667766 0.5226113 0.4159007  0.04 1 0.4692560
D1.0.05.12 0.4715247 0.02672273 0.5249702 0.4180793  0.05 1 0.4715247
                   se        U2        L2 order year
D1.12.12   0.02659434 0.5129960 0.4066186 first    0
D1.0.01.12 0.02660005 0.5154256 0.4090254 first    0
D1.0.02.12 0.02661603 0.5178385 0.4113743 first    0
D1.0.03.12 0.02664200 0.5202339 0.4136659 first    0
D1.0.04.12 0.02667766 0.5226113 0.4159007 first    0
D1.0.05.12 0.02672273 0.5249702 0.4180793 first    0
```

```r
xyplot(coef ~ month|order, ww) ####################  WORKED !!!!
```



Does the seasonal pattern change?

We can use an interaction between the seasonal periodic model and the secular model to address whether the seasonal pattern changes over time. To maintain parsimony the interaction can be constructed with a spline with fewer degree of freedom than

```
sp1d <- gspline(quintiles_with_crash, 1, 0)
fit_int <- lm(
  unemployment ~ sp3d(year) + per3(month) + year:per3(month),
  unemp)
summary(fit_int)
```

```
Call:
lm(formula = unemployment ~ sp3d(year) + per3(month) + year:per3(month),
    data = unemp)

Residuals:
     Min       1Q   Median       3Q      Max
-0.81947 -0.22455 -0.00629  0.21794  1.10521

Coefficients:
                             Estimate Std. Error t value Pr(>|t|)
(Intercept)                 7.188e+00  8.668e-02  82.928  < 2e-16 ***
sp3d(year)D1|0             -3.470e-01  2.366e-02 -14.666  < 2e-16 ***
sp3d(year)D2|0              4.750e-01  3.222e-02  14.742  < 2e-16 ***
```

```
sp3d(year)D3|0              -2.097e-01  1.448e-02 -14.480  < 2e-16 ***
sp3d(year)C3|-0.184         -5.304e-01  4.167e-02 -12.730  < 2e-16 ***
sp3d(year)C3|4.63            6.316e-01  5.122e-02  12.330  < 2e-16 ***
sp3d(year)C0|8.96            1.676e-01  4.539e-01   0.369 0.712182
sp3d(year)C1|8.96            1.105e+01  3.049e+00   3.625 0.000345 ***
sp3d(year)C2|8.96           -5.525e+01  1.275e+01  -4.334 2.07e-05 ***
sp3d(year)C3|8.96            1.095e+02  2.598e+01   4.215 3.41e-05 ***
sp3d(year)C3|9.45           -1.100e+02  2.599e+01  -4.231 3.19e-05 ***
sp3d(year)C3|14.3           -4.726e-02  6.572e-02  -0.719 0.472691
per3(month)D1|12/12          5.035e-01  3.680e-02  13.683  < 2e-16 ***
per3(month)D2|12/12          1.747e-01  7.832e-02   2.231 0.026517 *
per3(month)D3|12-/12        -8.550e-02  6.152e-02  -1.390 0.165779
per3(month)C3|2.4/2.4        8.184e-01  1.159e-01   7.062 1.39e-11 ***
per3(month)D1|12/12:year    -6.179e-03  3.674e-03  -1.682 0.093795 .
per3(month)D2|12/12:year     9.197e-03  7.863e-03   1.170 0.243167
per3(month)D3|12-/12:year    6.545e-03  6.184e-03   1.058 0.290850
per3(month)C3|2.4/2.4:year  7.824e-03  1.167e-02   0.671 0.503089
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3455 on 270 degrees of freedom
Multiple R-squared:  0.9175,    Adjusted R-squared:  0.9117
F-statistic:   158 on 19 and 270 DF,  p-value: < 2.2e-16
```

```r
car::Anova(fit_int)
```

```
Anova Table (Type II tests)

Response: unemployment
                  Sum Sq  Df  F value    Pr(>F)
sp3d(year)       203.433  11 154.8935 < 2.2e-16 ***
per3(month)       43.889   4  91.8964 < 2.2e-16 ***
per3(month):year   2.193   4   4.5919  0.001329 **
Residuals         32.237 270
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
wald(fit_int, ':')
```

```
  numDF denDF  F.value p.value
:     4   270 4.591883 0.00133

Coefficients                  Estimate Std.Error  DF   t-value p-value
  per3(month)D1|12/12:year   -0.006179  0.003674 270 -1.681638 0.09380
  per3(month)D2|12/12:year    0.009197  0.007863 270  1.169665 0.24317
  per3(month)D3|12-/12:year   0.006545  0.006184 270  1.058332 0.29085
  per3(month)C3|2.4/2.4:year  0.007824  0.011668 270  0.670537 0.50309

Coefficients                 Lower 0.95 Upper 0.95
  per3(month)D1|12/12:year    -0.013413   0.001055
  per3(month)D2|12/12:year    -0.006284   0.024678
  per3(month)D3|12-/12:year   -0.005630   0.018720
  per3(month)C3|2.4/2.4:year  -0.015149   0.030797
```

There is weak evidence of a change in the seasonal pattern, however, if we wished to visualize the seasonal pattern at different years we can proceed as follows.

```r
Lfx(fit_int)
```

```
list( 1,
1 * M(sp3d(year)),
1 * M(per3(month)),
1 * M(per3(month)) * year
)
```

```r
quintiles_with_crash
```

```
       20%        40%                  60%        80% 
-0.1839836  4.6340862  8.9555099  9.4483231 14.2642026 
```

```r
range(unemp$year)
```

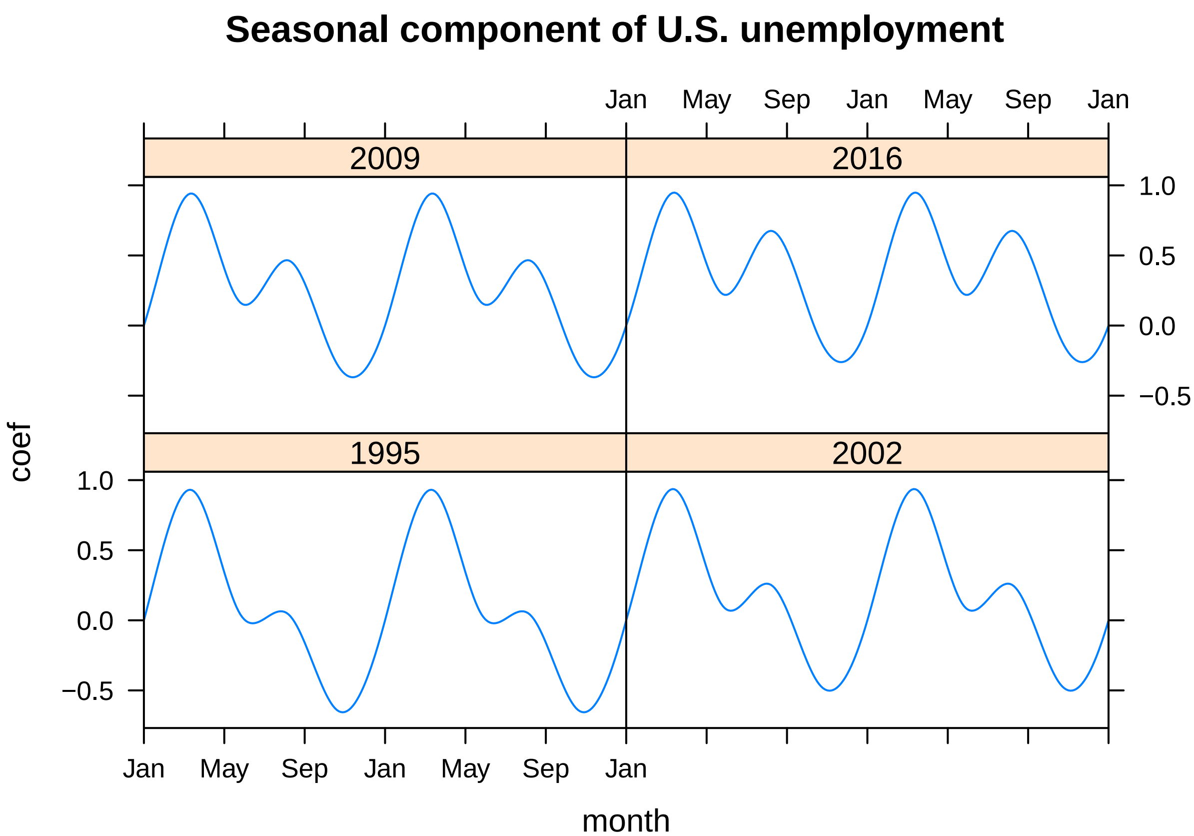
```
[1] -4.999316 19.085558
```

```r
pred <- expand.grid(
  month = seq(0,24,.1),
  date = as.Date(c('1995-01-01','2002-01-01','2009-01-01','2016-01-01')),
  D = 1)
pred$year <- toyear(pred$date)
ww <- walddf(
  fit_int,
  Lfx(fit_int,
      list( 0,
            0 * M(sp3d(year)),
            1 * M(per3(month)),
            1 * M(per3(month)) * year
      ), pred)
)
```

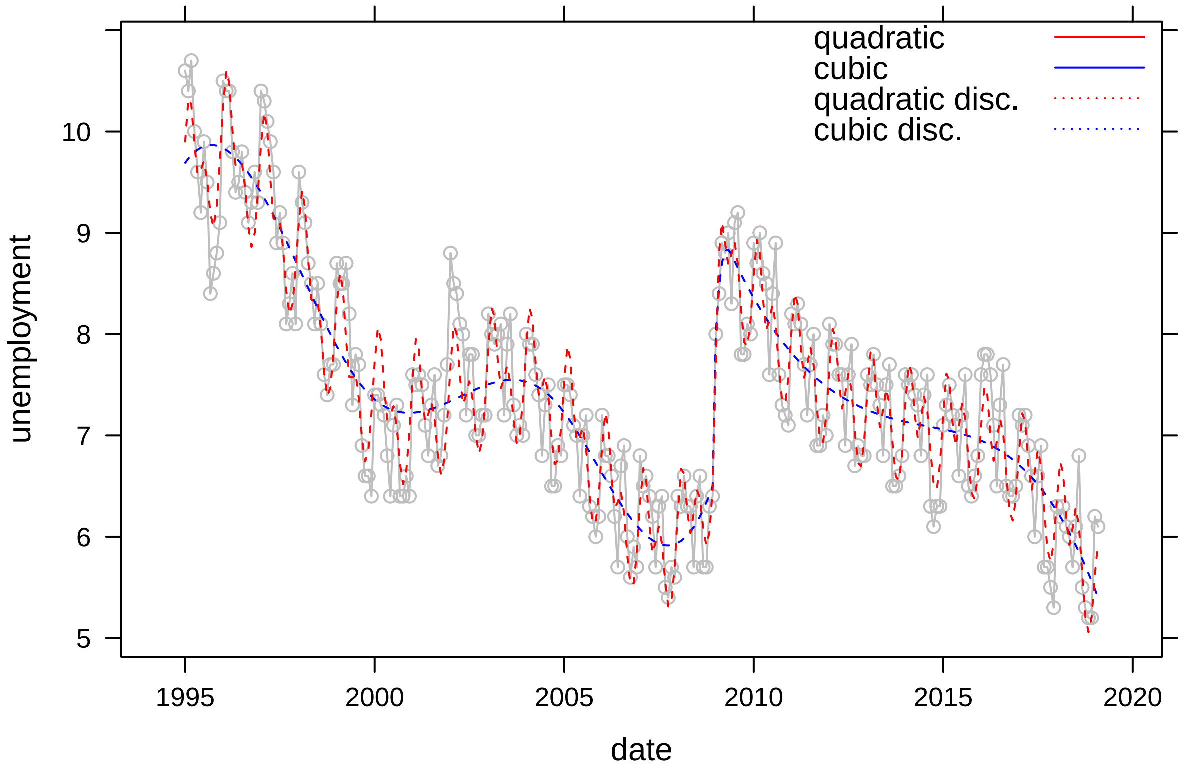
```
Warning in wald(fit = fit, Llist = Llist, clevel = clevel, data = data, :
Poorly conditioned L matrix, calculated numDF may be incorrect

Warning in min(dfs[x != 0]): no non-missing arguments to min; returning Inf

Warning in min(dfs[x != 0]): no non-missing arguments to min; returning Inf

Warning in min(dfs[x != 0]): no non-missing arguments to min; returning Inf

Warning in min(dfs[x != 0]): no non-missing arguments to min; returning Inf

Warning in min(dfs[x != 0]): no non-missing arguments to min; returning Inf

Warning in min(dfs[x != 0]): no non-missing arguments to min; returning Inf

Warning in min(dfs[x != 0]): no non-missing arguments to min; returning Inf

Warning in min(dfs[x != 0]): no non-missing arguments to min; returning Inf

Warning in min(dfs[x != 0]): no non-missing arguments to min; returning Inf

Warning in min(dfs[x != 0]): no non-missing arguments to min; returning Inf

Warning in min(dfs[x != 0]): no non-missing arguments to min; returning Inf
```

```
Warning in min(dfs[x != 0]): no non-missing arguments to min; returning Inf
```

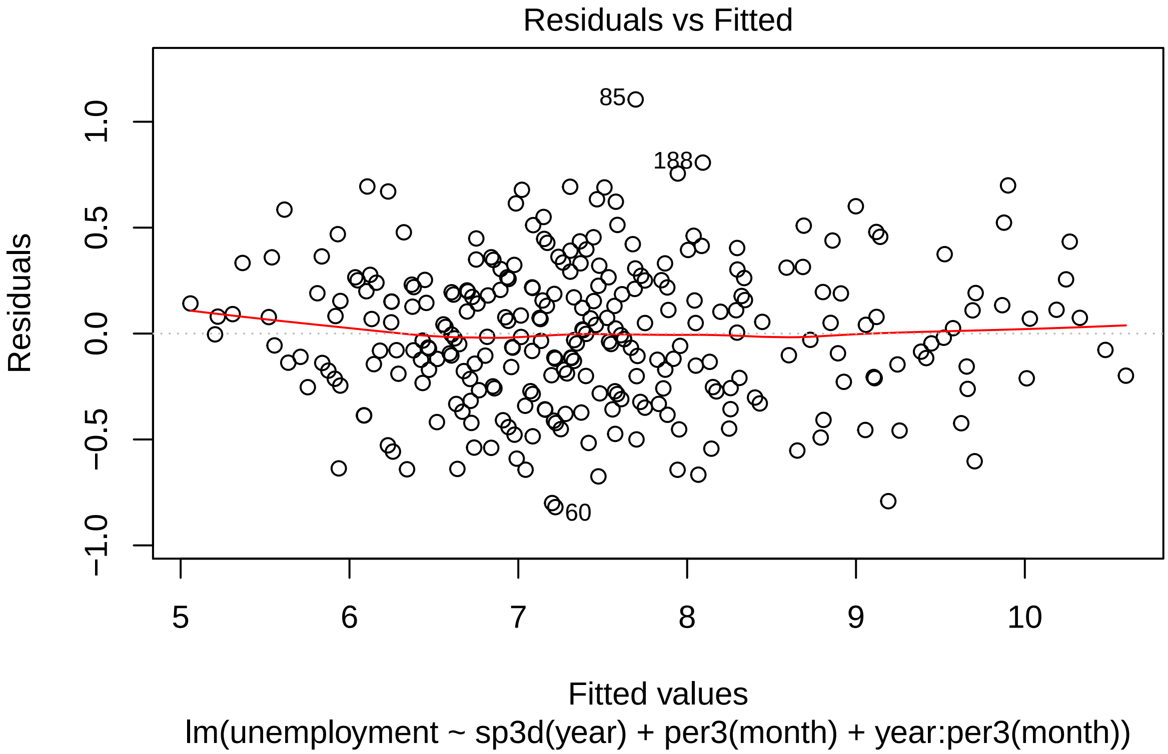
```
xyplot(coef ~ month | sub("-01-01","",date), ww,
       type = 'l',
       xlim = c(0,24),
       scales = list(
         x = list(
           at = c(0,4,8,12,16,20,24),
           labels = c('Jan','May','Sep','Jan','May','Sep','Jan'))),
       main = 'Seasonal component of U.S. unemployment')
```

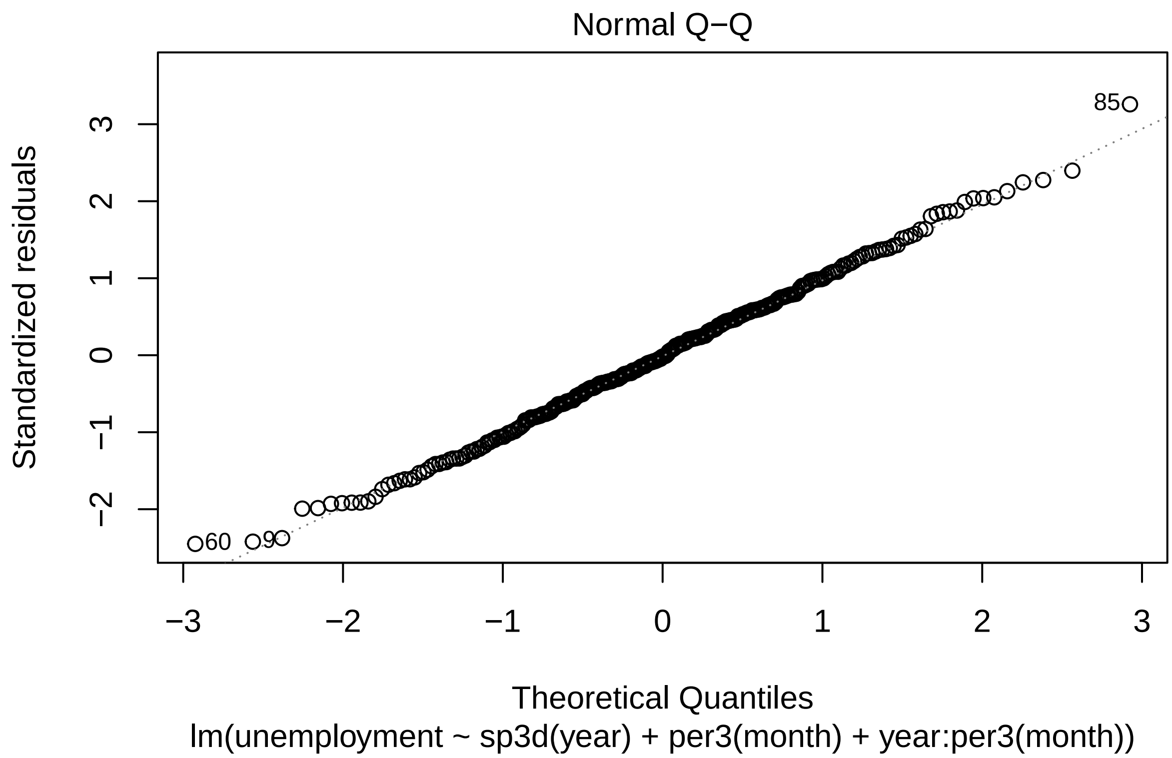


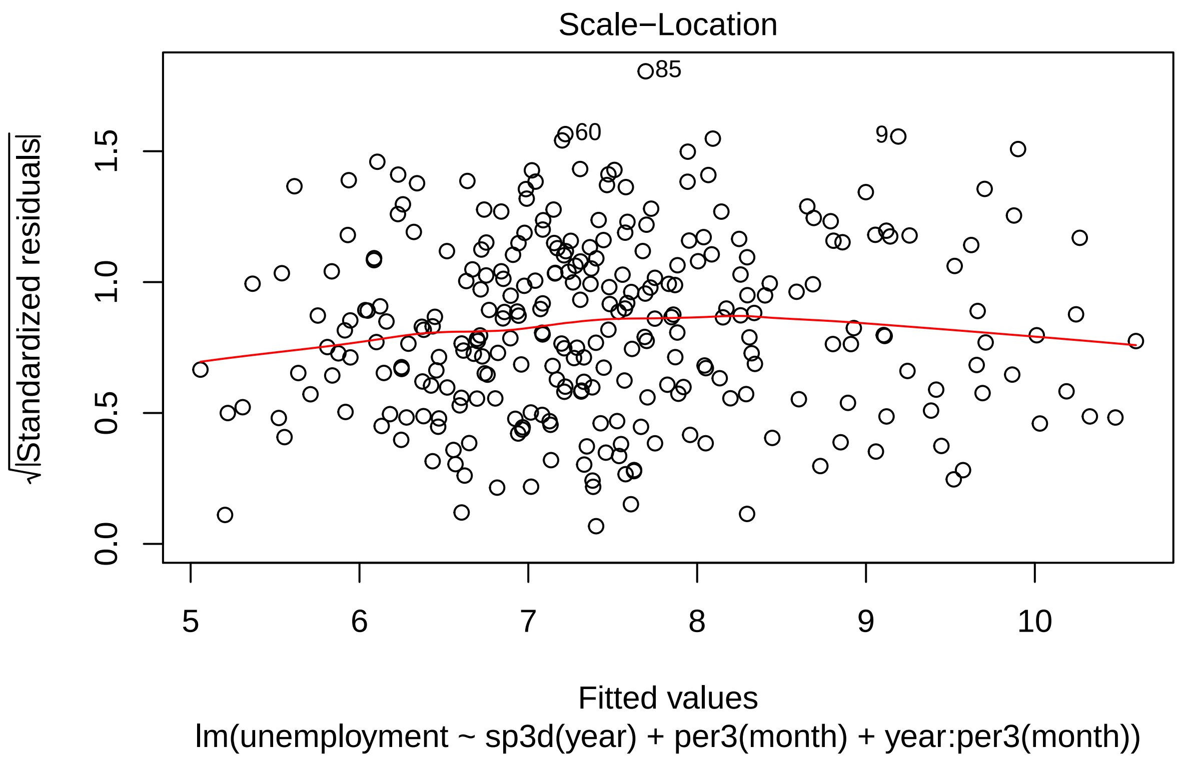

```
unemp$fit_int <- predict(fit_int, unemp)
###################################################################### REDO
pp <- xyplot(unemployment ~ date, unemp, type = 'b',
             col = 'gray',
             key = list(
               corner = c(1,1),
               text = list(lab = c('quadratic','cubic','quadratic disc.','cubic disc.')),
               lines = list(col= c('red','blue','red','blue'),
                            lty = c(1,1,3,3))
               )) +
  layer(panel.lines(x, unemp$fit3d, col = 'blue', lty = 2)) +
  layer(panel.lines(x, unemp$fit_int, col = 'red', lty = 2))
pp
```

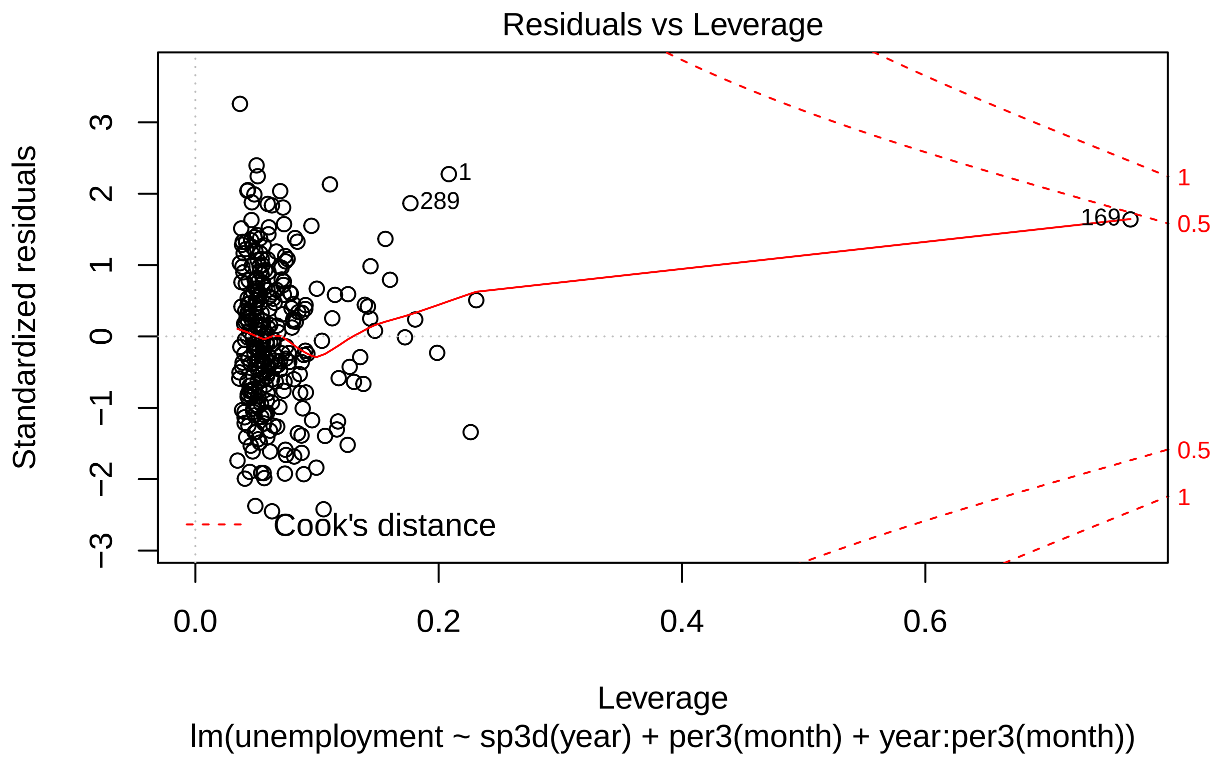



```r
# diagnostics
plot(fit_int)
```


Residuals vs Fitted
85
188
60
Residuals
1.0
0.5
0.0
−0.5
−1.0
5
6
7
8
9
10
Fitted values
lm(unemployment ~ sp3d(year) + per3(month) + year:per3(month))


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(unemployment ~ sp3d(year) + per3(month) + year:per3(month))
85
60
9
−3
−2
−1
0
1
2
3


Scale−Location
85
60
9
1.5
1.0
0.5
0.0
√|Standardized residuals|
5
6
7
8
9
10
Fitted values
lm(unemployment ~ sp3d(year) + per3(month) + year:per3(month))




```
unemp[169,]  # as expected
```

```
          date unemployment month     year     fit2     fit3    fit2d
169 2009-01-01            8     1 9.002053 7.081329 7.186382 8.149947
       fit3d  fitper3  fit_int
169 8.016123 7.717591 7.727456
```

```
library(nlme)
fit_int_ar <-
  gls(
    unemployment ~ sp3d(year) + per3(month) + year:per3(month),
    unemp, correlation = corAR1(form = ~ 1))
summary(fit_int_ar)
```

```
Generalized least squares fit by REML
  Model: unemployment ~ sp3d(year) + per3(month) + year:per3(month)
  Data: unemp
      AIC      BIC    logLik
  347.239 426.4043 -151.6195

Correlation Structure: AR(1)
 Formula: ~1
 Parameter estimate(s):
      Phi
0.2061011

Coefficients:
```

```
                            Value Std.Error   t-value p-value
(Intercept)                7.20327  0.102450  70.30987  0.0000
sp3d(year)D1|0            -0.34489  0.029310 -11.76710  0.0000
sp3d(year)D2|0             0.47041  0.040032  11.75086  0.0000
sp3d(year)D3|0            -0.20776  0.017987 -11.55040  0.0000
sp3d(year)C3|-0.184       -0.52275  0.051546 -10.14142  0.0000
sp3d(year)C3|4.63          0.62486  0.063263   9.87704  0.0000
sp3d(year)C0|8.96          0.34431  0.479463   0.71813  0.4733
sp3d(year)C1|8.96          9.98363  3.360159   2.97118  0.0032
sp3d(year)C2|8.96        -50.71109 14.007588  -3.62026  0.0004
sp3d(year)C3|8.96        100.30656 28.586731   3.50885  0.0005
sp3d(year)C3|9.45       -100.76493 28.602630  -3.52293  0.0005
sp3d(year)C3|14.3         -0.05005  0.081099  -0.61720  0.5376
per3(month)D1|12/12        0.49929  0.041326  12.08179  0.0000
per3(month)D2|12/12        0.15848  0.084733   1.87031  0.0625
per3(month)D3|12-/12      -0.09534  0.066487  -1.43398  0.1527
per3(month)C3|2.4/2.4      0.79264  0.125229   6.32950  0.0000
per3(month)D1|12/12:year  -0.00603  0.004122  -1.46251  0.1448
per3(month)D2|12/12:year   0.00939  0.008493   1.10514  0.2701
per3(month)D3|12-/12:year  0.00658  0.006677   0.98484  0.3256
per3(month)C3|2.4/2.4:year 0.00803  0.012591   0.63808  0.5240

 Correlation:
                          (Intr) s3()D1 s3()D2 s3()D3 s3()C3|- s3()C3|4
sp3d(year)D1|0            -0.034
sp3d(year)D2|0            -0.523 -0.368
sp3d(year)D3|0             0.488  0.129 -0.956
sp3d(year)C3|-0.184        0.443  0.457 -0.974  0.921
sp3d(year)C3|4.63         -0.362  0.087  0.763 -0.902 -0.742
sp3d(year)C0|8.96          0.015 -0.031 -0.076  0.109  0.079   -0.194
sp3d(year)C1|8.96          0.047 -0.019 -0.051  0.074  0.052   -0.118
sp3d(year)C2|8.96         -0.031  0.005  0.012 -0.019 -0.012    0.042
sp3d(year)C3|8.96          0.034 -0.007 -0.020  0.030  0.020   -0.055
sp3d(year)C3|9.45         -0.034  0.007  0.019 -0.028 -0.019    0.053
sp3d(year)C3|14.3          0.022 -0.011 -0.001  0.002  0.001   -0.004
per3(month)D1|12/12       -0.023 -0.053  0.063 -0.055 -0.084    0.048
per3(month)D2|12/12       -0.599  0.145  0.010 -0.011 -0.014    0.015
per3(month)D3|12-/12      -0.516  0.137 -0.005  0.002  0.006    0.002
per3(month)C3|2.4/2.4     -0.422  0.080  0.030 -0.027 -0.040    0.026
per3(month)D1|12/12:year   0.019  0.051 -0.048  0.037  0.067   -0.020
per3(month)D2|12/12:year   0.425 -0.201 -0.002  0.000  0.005    0.004
per3(month)D3|12-/12:year  0.365 -0.185  0.008 -0.007 -0.009    0.007
per3(month)C3|2.4/2.4:year 0.300 -0.119 -0.019  0.014  0.028   -0.006
                          s3()C0 s3()C1 s3()C2 s3()C3|8 s3()C3|9 s3()C3|1
sp3d(year)D1|0
sp3d(year)D2|0
sp3d(year)D3|0
sp3d(year)C3|-0.184
sp3d(year)C3|4.63
sp3d(year)C0|8.96
sp3d(year)C1|8.96         -0.838
sp3d(year)C2|8.96          0.852 -0.995
sp3d(year)C3|8.96         -0.846  0.995 -1.000
sp3d(year)C3|9.45          0.846 -0.995  1.000 -1.000
```

```
sp3d(year)C3|14.3          -0.134  0.317 -0.360  0.366   -0.367
per3(month)D1|12/12        -0.063  0.040 -0.043  0.042   -0.042    0.007
per3(month)D2|12/12         0.000 -0.013  0.012 -0.013    0.013   -0.013
per3(month)D3|12-/12        0.013 -0.019  0.019 -0.019    0.019   -0.012
per3(month)C3|2.4/2.4      -0.034  0.019 -0.021  0.020   -0.020    0.003
per3(month)D1|12/12:year   -0.028  0.021 -0.019  0.019   -0.019   -0.030
per3(month)D2|12/12:year   -0.003 -0.003  0.003 -0.003    0.003    0.009
per3(month)D3|12-/12:year   0.003 -0.007  0.006 -0.006    0.006    0.012
per3(month)C3|2.4/2.4:year -0.018  0.012 -0.011  0.011   -0.011   -0.019
                           pr3()D1|12/12 pr3()D2|12/12 pr3()D3|12-/12
sp3d(year)D1|0
sp3d(year)D2|0
sp3d(year)D3|0
sp3d(year)C3|-0.184
sp3d(year)C3|4.63
sp3d(year)C0|8.96
sp3d(year)C1|8.96
sp3d(year)C2|8.96
sp3d(year)C3|8.96
sp3d(year)C3|9.45
sp3d(year)C3|14.3
per3(month)D1|12/12
per3(month)D2|12/12        -0.006
per3(month)D3|12-/12       -0.306         0.941
per3(month)C3|2.4/2.4       0.494         0.793         0.578
per3(month)D1|12/12:year   -0.711         0.004         0.219
per3(month)D2|12/12:year    0.008        -0.714        -0.673
per3(month)D3|12-/12:year   0.220        -0.670        -0.714
per3(month)C3|2.4/2.4:year -0.345        -0.565        -0.413
                           pr3()C3|2.4/2.4 p3()D1|12/12: p3()D2|12/12:
sp3d(year)D1|0
sp3d(year)D2|0
sp3d(year)D3|0
sp3d(year)C3|-0.184
sp3d(year)C3|4.63
sp3d(year)C0|8.96
sp3d(year)C1|8.96
sp3d(year)C2|8.96
sp3d(year)C3|8.96
sp3d(year)C3|9.45
sp3d(year)C3|14.3
per3(month)D1|12/12
per3(month)D2|12/12
per3(month)D3|12-/12
per3(month)C3|2.4/2.4
per3(month)D1|12/12:year   -0.350
per3(month)D2|12/12:year   -0.565           -0.005
per3(month)D3|12-/12:year  -0.411           -0.304         0.941
per3(month)C3|2.4/2.4:year -0.710            0.494         0.790
                           p3()D3|12-/12:
sp3d(year)D1|0
sp3d(year)D2|0
sp3d(year)D3|0
sp3d(year)C3|-0.184
```

```
sp3d(year)C3|4.63
sp3d(year)C0|8.96
sp3d(year)C1|8.96
sp3d(year)C2|8.96
sp3d(year)C3|8.96
sp3d(year)C3|9.45
sp3d(year)C3|14.3
per3(month)D1|12/12
per3(month)D2|12/12
per3(month)D3|12-/12
per3(month)C3|2.4/2.4
per3(month)D1|12/12:year
per3(month)D2|12/12:year
per3(month)D3|12-/12:year
per3(month)C3|2.4/2.4:year  0.575

Standardized residuals:
        Min          Q1         Med          Q3         Max
-2.37792460 -0.60362972 -0.02779245  0.61281912  3.14160864

Residual standard error: 0.3508346
Degrees of freedom: 290 total; 270 residual
```

```r
anova(fit_int_ar, fit_int)
```

```
           Model df      AIC      BIC    logLik   Test  L.Ratio p-value
fit_int_ar     1 22 347.2390 426.4043 -151.6195
fit_int        2 21 355.9367 431.5036 -156.9684 1 vs 2 10.69773  0.0011
```

```r
car::Anova(fit_int_ar)
```

```
Analysis of Deviance Table (Type II tests)

Response: unemployment
                 Df    Chisq Pr(>Chisq)
sp3d(year)       11 1127.600    < 2e-16 ***
per3(month)       4  274.712    < 2e-16 ***
per3(month):year  4   13.017    0.01119 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
wald(fit_int, ':')
```

```
  numDF denDF  F.value p.value
:     4   270 4.591883 0.00133

Coefficients                    Estimate Std.Error  DF   t-value p-value
  per3(month)D1|12/12:year   -0.006179  0.003674 270 -1.681638 0.09380
  per3(month)D2|12/12:year    0.009197  0.007863 270  1.169665 0.24317
  per3(month)D3|12-/12:year   0.006545  0.006184 270  1.058332 0.29085
  per3(month)C3|2.4/2.4:year  0.007824  0.011668 270  0.670537 0.50309

Coefficients                 Lower 0.95 Upper 0.95
  per3(month)D1|12/12:year    -0.013413   0.001055
  per3(month)D2|12/12:year    -0.006284   0.024678
  per3(month)D3|12-/12:year   -0.005630   0.018720
```

```
  per3(month)C3|2.4/2.4:year  -0.015149   0.030797
```

```r
fit_int_ar2 <- update(fit_int_ar, correlation = corARMA(form = ~ 1, p = 2, q = 0))
anova( fit_int_ar2, fit_int_ar, fit_int)
```

```
            Model df      AIC      BIC    logLik   Test   L.Ratio p-value
fit_int_ar2     1 23 345.5980 428.3617 -149.7990
fit_int_ar      2 22 347.2390 426.4043 -151.6195 1 vs 2  3.641013  0.0564
fit_int         3 21 355.9367 431.5036 -156.9684 2 vs 3 10.697733  0.0011
```

Comparison with a Fourier model

```r
Sin <- function(x) cbind(sin=sin(2*pi*x),cos=cos(2*pi*x))
fit_fourier_int <- gls(
  unemployment ~ sp3d(year) + Sin(month/12) + Sin(2*month/12) + Sin(3*month/12)+
    Sin(4*month/12) +
    year:(Sin(month/12) + Sin(2*month/12)),
  unemp, correlation = corAR1(form = ~ 1))
car::Anova(fit_fourier_int)
```

```
Analysis of Deviance Table (Type II tests)

Response: unemployment
                      Df     Chisq Pr(>Chisq)
sp3d(year)            11 1311.0913  < 2.2e-16 ***
Sin(month/12)          2  161.9144  < 2.2e-16 ***
Sin(2 * month/12)      2  216.2328  < 2.2e-16 ***
Sin(3 * month/12)      2   57.3536  3.514e-13 ***
Sin(4 * month/12)      2  212.2903  < 2.2e-16 ***
Sin(month/12):year     2   12.9711   0.001525 **
Sin(2 * month/12):year  2    2.5628   0.277646
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
AIC(fit_fourier_int, fit_int_ar)
```

```
Warning in AIC.default(fit_fourier_int, fit_int_ar): models are not all
fitted to the same number of observations

                df     AIC
fit_fourier_int 26 194.148
fit_int_ar      22 347.239
```

```r
fit_factor_int <- lm(
  unemployment ~ sp3d(year) + factor(month) + year:factor(month),
  unemp)

car::Anova(fit_factor_int)
```

```
Anova Table (Type II tests)

Response: unemployment
                    Sum Sq  Df  F value    Pr(>F)
sp3d(year)         173.203  10 280.6698 < 2.2e-16 ***
factor(month)       59.481  11  87.6251 < 2.2e-16 ***
factor(month):year   3.040  11   4.4785 3.461e-06 ***
```

```
Residuals           15.798 256
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(fit_factor_int, fit_int)
```

```
Analysis of Variance Table

Model 1: unemployment ~ sp3d(year) + factor(month) + year:factor(month)
Model 2: unemployment ~ sp3d(year) + per3(month) + year:per3(month)
  Res.Df    RSS  Df Sum of Sq      F    Pr(>F)
1    256 15.798
2    270 32.237 -14   -16.439 19.028 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

References to incorporate

- Spline derivatives

“‘

References to incorporate

- Spline derivatives