STAT 425

Regression Splines

Regression Splines

Recall that for a set of given knots, the corresponding cubic splines form a linear space (of functions) with dim $(m+4)$.
Regression splines uses a basis expansion approach:

$$g(x) = \beta_1 h_1(x) + \beta_2 h_2(x) + \ldots + \beta_p h_p(x)$$

- If cubic splines are used as basis functions $p = m + 4$
- If Natural Cubic Splines (NCS) are used as basis functions $p = m$

We can represent the model on the observed n data points using matrix notation:

$$\begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{pmatrix}_{n\times 1} = \begin{pmatrix} h_1(x_1) & h_2(x_1) & \dots & h_p(x_1) \\ h_1(x_2) & h_2(x_2) & \dots & h_p(x_2) \\ & & & \\ h_1(x_n) & h_2(x_n) & \dots & h_p(x_n) \end{pmatrix}_{n\times p} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \dots \\ \beta_n \end{pmatrix}_{p\times 1}$$

Our design matrix is the matrix $\mathbf{F}$ of basis functions. We can find $\boldsymbol{\beta}$ by solving the problem:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} ||\mathbf{y} - \mathbf{F}\boldsymbol{\beta}||^2$$

Regression Splines in **R**

- We can obtain the design matrix $\mathbf{F}$ by commands bs (B-splines) or ns (NCS) in **R**, and then call the regression function lm.
- To select the number of knots we can use K-fold cross-validation (CV) (More on this later).

B-splines basis functions

Understand how **R** counts the degrees-of-freedom

- To generate a B-spline basis for a given set of x_i's, you can use the command bs.
- You can tell **R**, the location of knots.
- Or you can tell **R** the df. Recall that a cubic spline with m knots has $m + 4$ df, so we need $m = df - 4$ knots.
- By default, **R** puts knots at the $1/(m+1), \ldots, m/(m+1)$ quantiles of $x_{1:n}$.

Example: B-splines with 5 knots (9 df)

Define knots in the interval $[0, 2]$ and get matrix $\mathbf{F}$

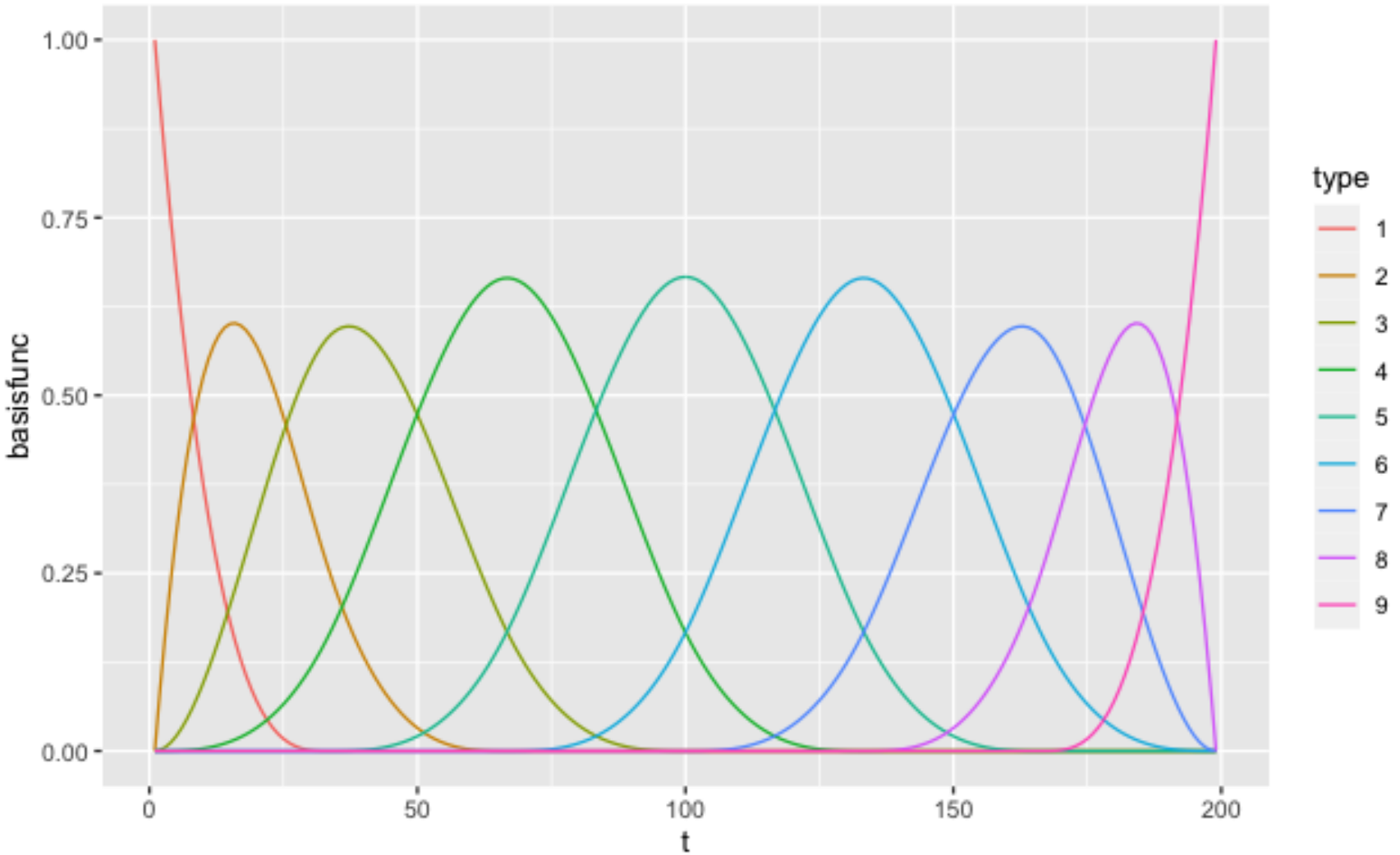
```
 library(splines)
 x=(1:199)/100
 n = length(x)
 m=5
 myknots= 2*(1:m)/(m+1)
 myknots
```

```
## [1] 0.3333333 0.6666667 1.0000000 1.3333333 1.6666667
```

```
 #Using the Intercept option
 F=bs(x,knots=myknots, intercept=TRUE)
 dim(F)
```

```
## [1] 199   9
```

B-splines basis functions with df=9 (knots=5)



- How **R** counts the df might be confusing. The df in command bs actually means the number of columns of the design matrix returned by bs. So if the intercept is not included in the design matrix (which is the default), then the df in command bs is equal to the real df minus 1.

```
 #No Intercept option
 F=bs(x,knots=myknots)
 dim(F)

## [1] 199   8
```

The following three design matrices (the first two are of $n \times 5$ and the last one is of $n \times 6$) correspond to the same regression model with cubic splines of df=6.

```
 F1=bs(x, knots=quantile(x, c(1/3, 2/3)))
 dim(F1)

## [1] 199   5

 F2=bs(x, df=5)
 dim(F2)

## [1] 199   5

 F3=bs(x, df=6, intercept=TRUE)
 dim(F3)

## [1] 199   6
```

NCS Basis functions

- To generate a NCS basis for a given set of x_i's, use the command ns.
- Recall that the linear functions in the two extreme intervals are totally determined by the other cubic splines. So data points in the two extreme intervals (i.e., outside the two boundary knots) are wasted since they do not affect the fitting. Therefore, by default, **R** puts the two boundary knots as the min and max of the x_i's
- You can tell **R** the location of knots, which are the interior knots. Recall that a NCS with m knots has m df. So the df is equal to the number of (interior) knots plus 2, where 2 means the two boundary knots.

- Or you can tell **R** the df. If *intercept* $=$ *TRUE*, then we need $m = df - 2$ knots, otherwise we need $m = df - 1$ knots. Again, by default, **R** puts knots at the $1/(m+1), ..., m/(m+1)$ quantiles of $x_{1:n}$.
- The following three design matrices (the first two are of $n \times 3$ and the last one is of $n \times 4$) correspond to the same regression model with NCS of $df = 4$.

```
 F1=ns(x, knots=quantile(x, c(1/3, 2/3)))
 dim(F1)
```

```
## [1] 199   3
```

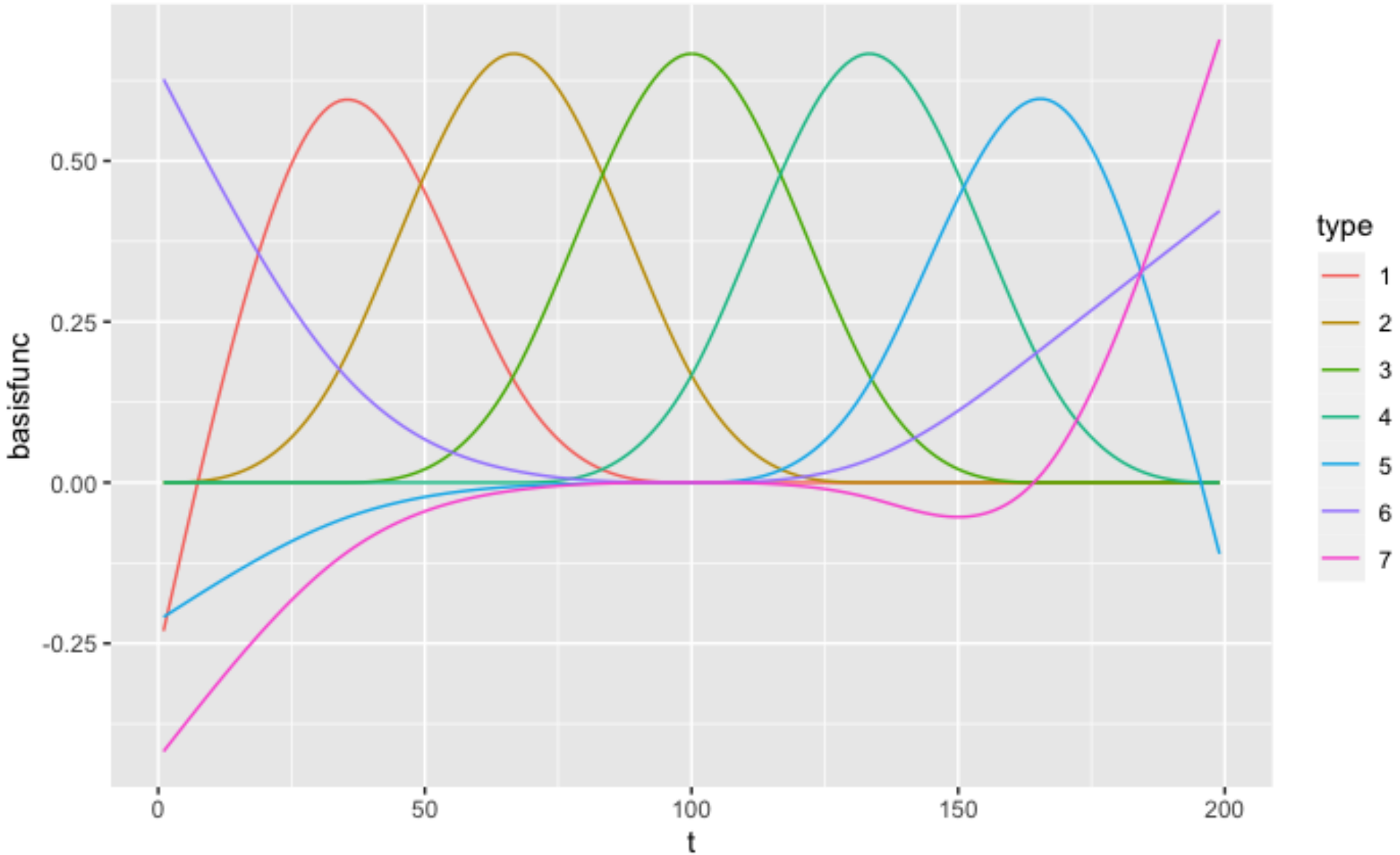
```
 F2=ns(x, df=3)
 dim(F2)
```

```
## [1] 199   3
```

```
 F3=ns(x, df=4, intercept=TRUE)
 dim(F3)
```

```
## [1] 199   4
```

Natural cubic splines basis functions with df=7 (knots=5)



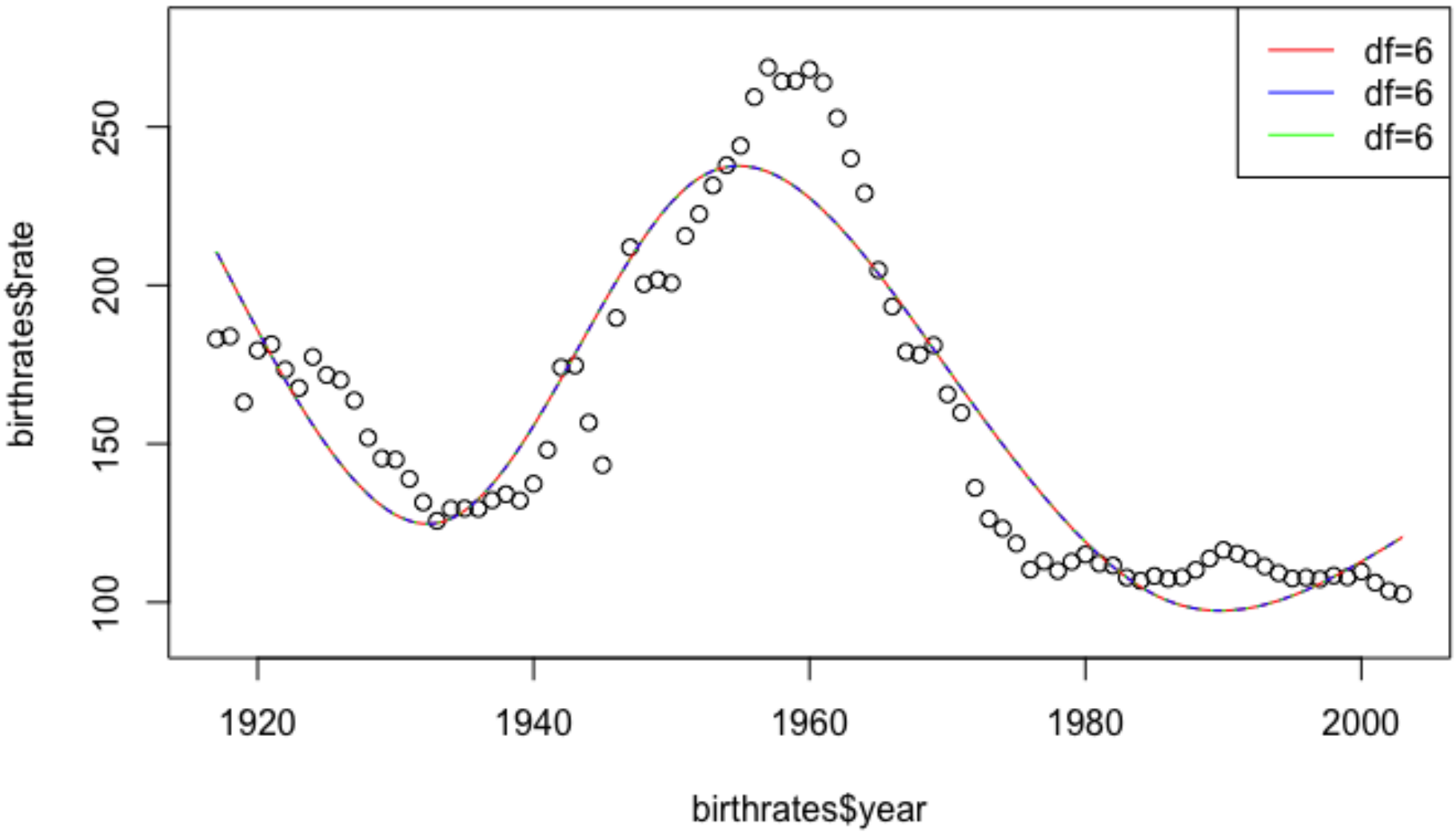
Example: Birth rate data set

This data set lists the number of live births per 10,000 23-year-old women in the United States between 1917 and 2003.
Use NCS with number of knots=4 and df=6 (including the two boundary knots)

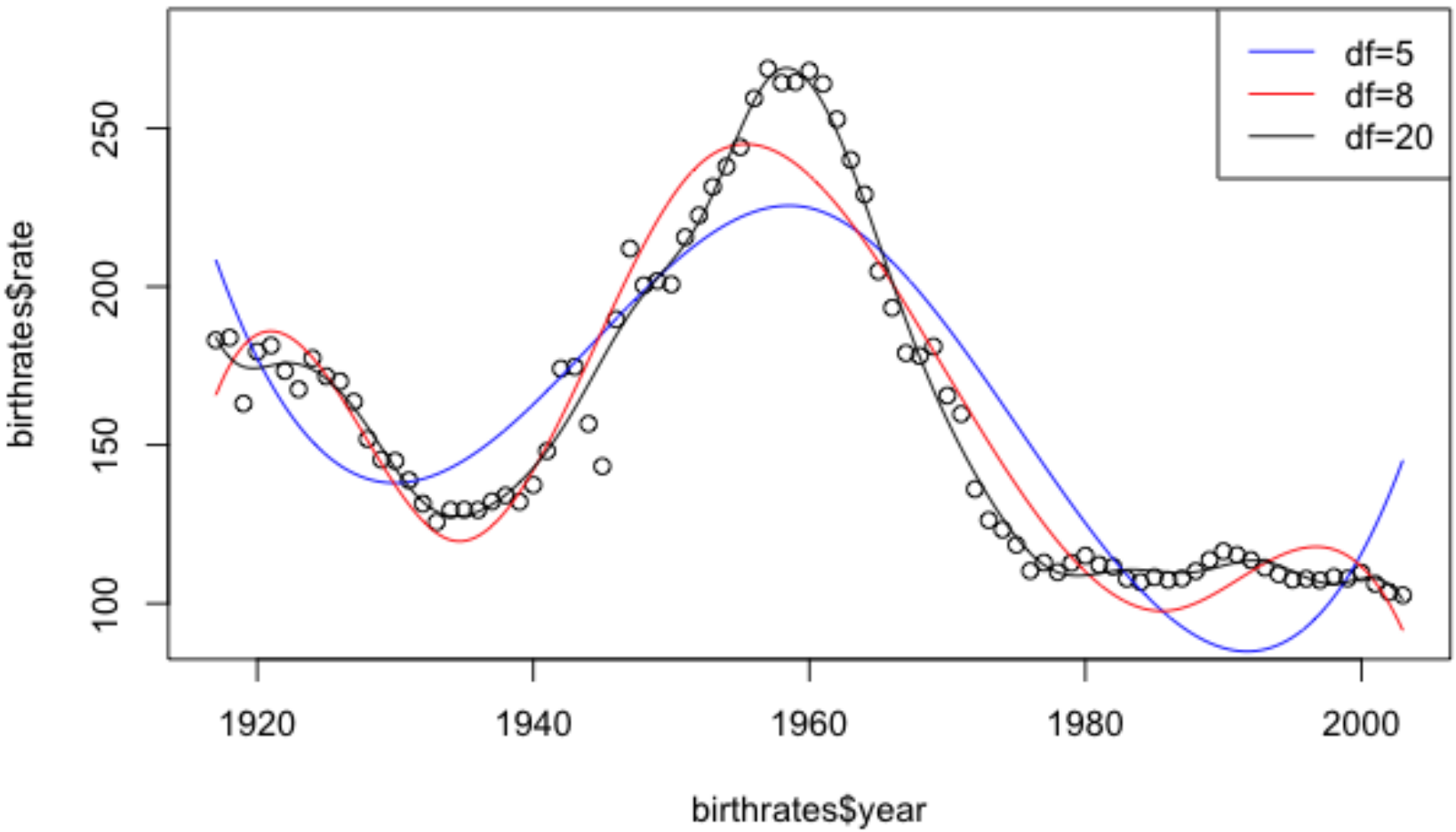
```
source("birthrates.txt");
birthrates = as.data.frame(birthrates)
names(birthrates) = c("year", "rate")
# The following fitted models provide the same results
fit1=lm(rate~ns(year, knots=quantile(year, (1:4)/5)),
data=birthrates);
fit2=lm(rate~ns(year, df=5), data=birthrates);
fit3=lm(rate~ns(year, df=6, intercept=TRUE),
data=birthrates)
```

Birth rate example

All fitted models provide the same results. However the number of knots might not be optimal.



Comparing different number of knots in the birth rate example



A good way to select the optimal number of knots (or df) is to use K-fold Cross-Validation:

1. Set a fixed number of knots (or df)
2. Divide the set of observations into k groups (or *folds*)
3. Leave the first fold as a validation set (not used to fit the model). Fit the Regression Spline with a fixed number of knots using the remaining $k-1$ folds.
4. Calculate the Mean Square Error for fold 1: MSE_1
5. Repeat the previous steps k times. Each time a new validation set is used to calculate MSE_i
6. Calculate the average k-fold Cross-Validation error: $CV(k) = \frac{1}{k}\sum_{i=1}^{k} MSE_i$
7. Repeat 2 to 6 with a new number of knots (or df)
8. Select the number of knots that minimizes the k-fold CV error or $CV(k)$