Smoothing Splines

Dan Kelley, Dalhousie University

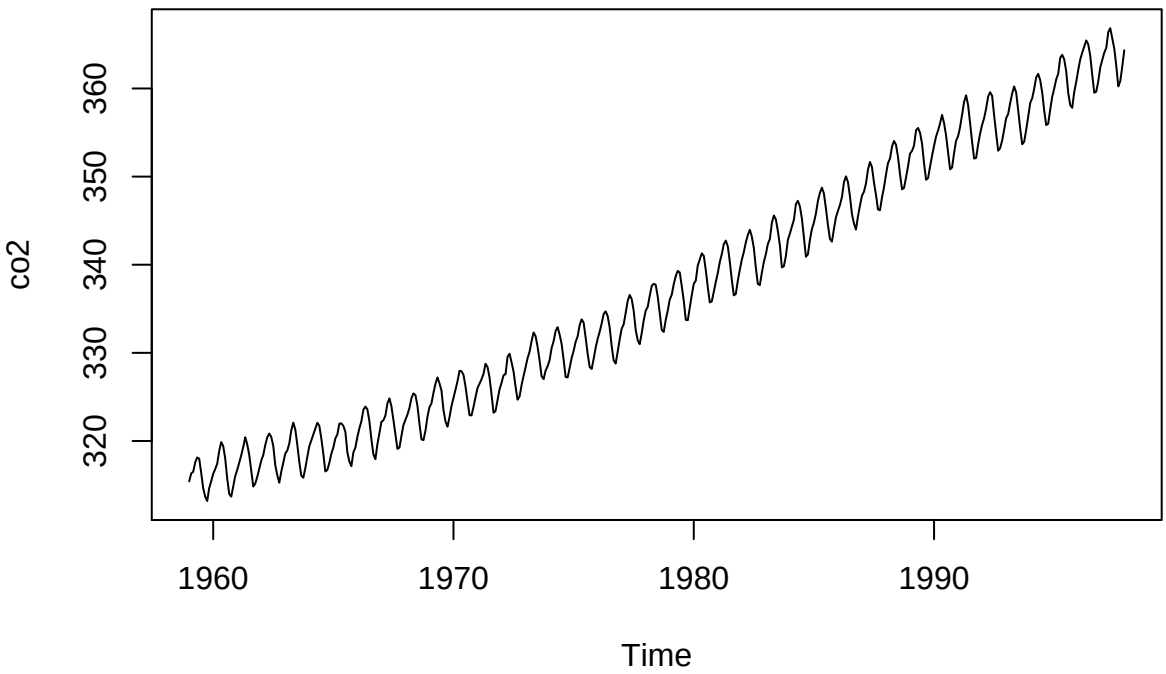
Goal. Demonstrated the use of `smooth.spline` with the built-in `co2` dataset.

This is a built-in dataset, so the next line is not required. However, it doesn't hurt :-)

```
data(co2)
```

First, plot the dataset.

```
plot(co2)
```



Note: `co2` is a time-series object (i.e. it inherits from `ts`), so `plot()` knows how to handle it. Still, let's extract the time, for convenience.

```
t <- time(co2)
```

Now, it's time to get down to the business of fitting a smoothing spline. If we call `smooth.spline()` with just `t` and `co2` as arguments, it will do a clever analysis of the data and select the parameter for smoothing. In many cases, this is the desired effect, but for now, let's imagine we are hoping to smooth over the seasonal variability, while still retaining some variation on decadal scales. There are several ways to set up `smooth.spline` to do this, one of which is to set the `df` parameter (thus setting the number of degrees of freedom). One should read the documentation (`?smooth.spline`) to learn how to do this systematically, but it is also helpful to build intuition by trying some values.

First, note that we have

```
length(t)
```

```
## [1] 468
```

samples in the data set, representing sampling at a rate of

```
diff(t[1:2])
```

```
## [1] 0.08333333
```

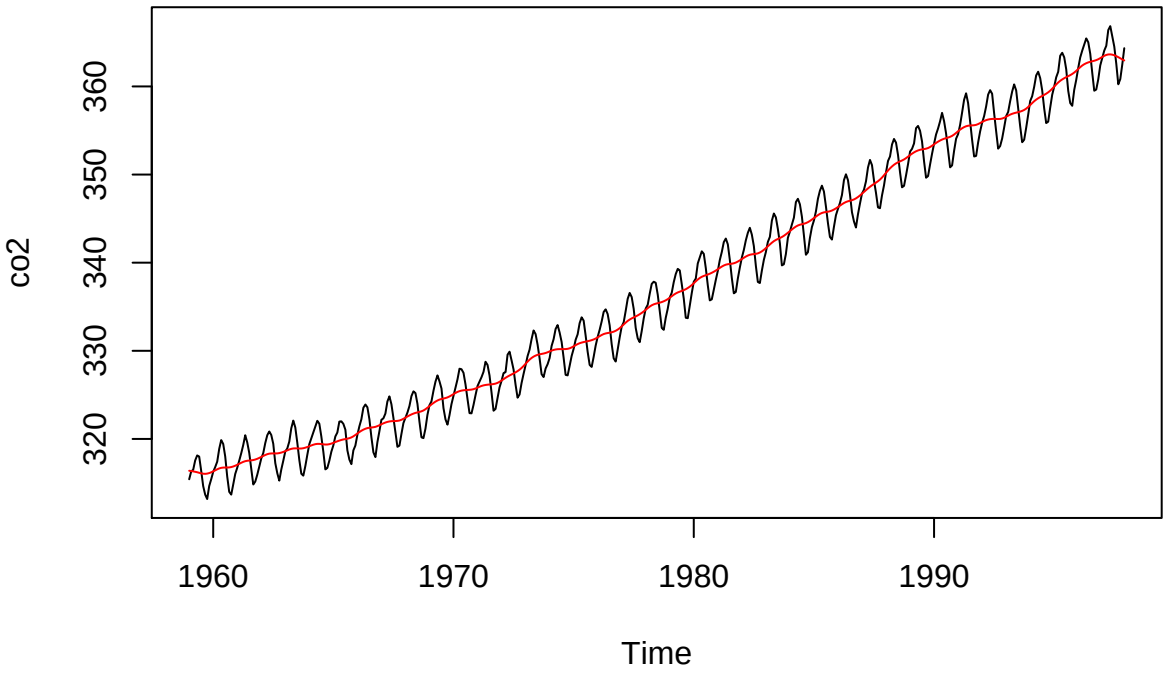
per year, i.e. one datum every month. Roughly speaking, we could smooth over a year by setting 1 degree of freedom per year, i.e. by setting `df` to

```
length(t) * diff(t[1:2])
```

```
## [1] 39
```

or, roughly, 40. Let's try that, plotting the spline on top of the data.

```
plot(co2)
S <- smooth.spline(t, co2, df=40)
lines(S$x, S$y, col='red')
```



(See `?smooth.spline` to learn that the output contains `x` and `y` ... and note that it contains also a wealth of useful information on the spline fit, itself.)

The graph illustrates that the `df` value retains some interannual variability, while removing most seasonal variability. Whether the smoothing is sufficient (or excessive) depends on the purpose at hand (see exercises).

Pay particular attention to the endpoints. The behaviour there may be undesired. (This is a general hint in all time-series work – look closely at the endpoints!)

Exercises.

1. Plot the residual, i.e. `co2-S$y` and think about its meaning (e.g. why is 1992 anomalous?)

2. Plot the spline prediction alone, and try different `df` values to get the smoothness you think you need, for some purpose of interest to you.

3. Apply similar methods to CTD profiles, e.g. extracting data from an `oce` dataset named `ctd`, or from one of the hundred-odd stations in the `oce` dataset named `section`.

4. **Advanced.** Read about the `deriv` argument of the `predict` method for smoothing splines (find this by typing `?predict.smooth.spline` in a console) and use what you've learned to calculate $N^2 = -(g/\rho_0)\partial\rho/\partial z$ for a CTD profile[1], adjusting `df` based on criteria of your own selection, noting how N^2 varies. NB. the `oce` function named `swN2` calculates N^2 in this way, setting `df` based on the number of data, etc.

[1] You might want to use a profile from exercise 2 or 3, or you might want to construct one in R ... or you might want to ask the workshop leaders how to read a `.cnv` file on your computer!