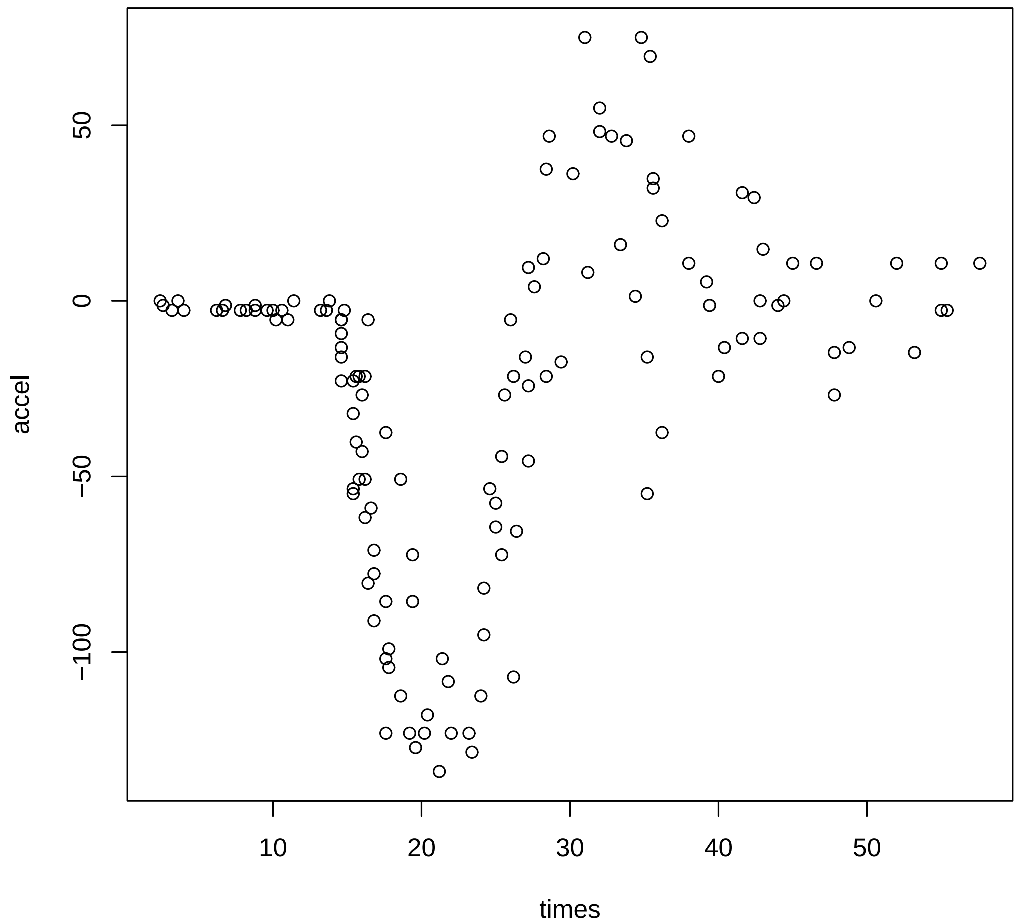
STAT 441: Lecture 18
More regression: flexible modeling

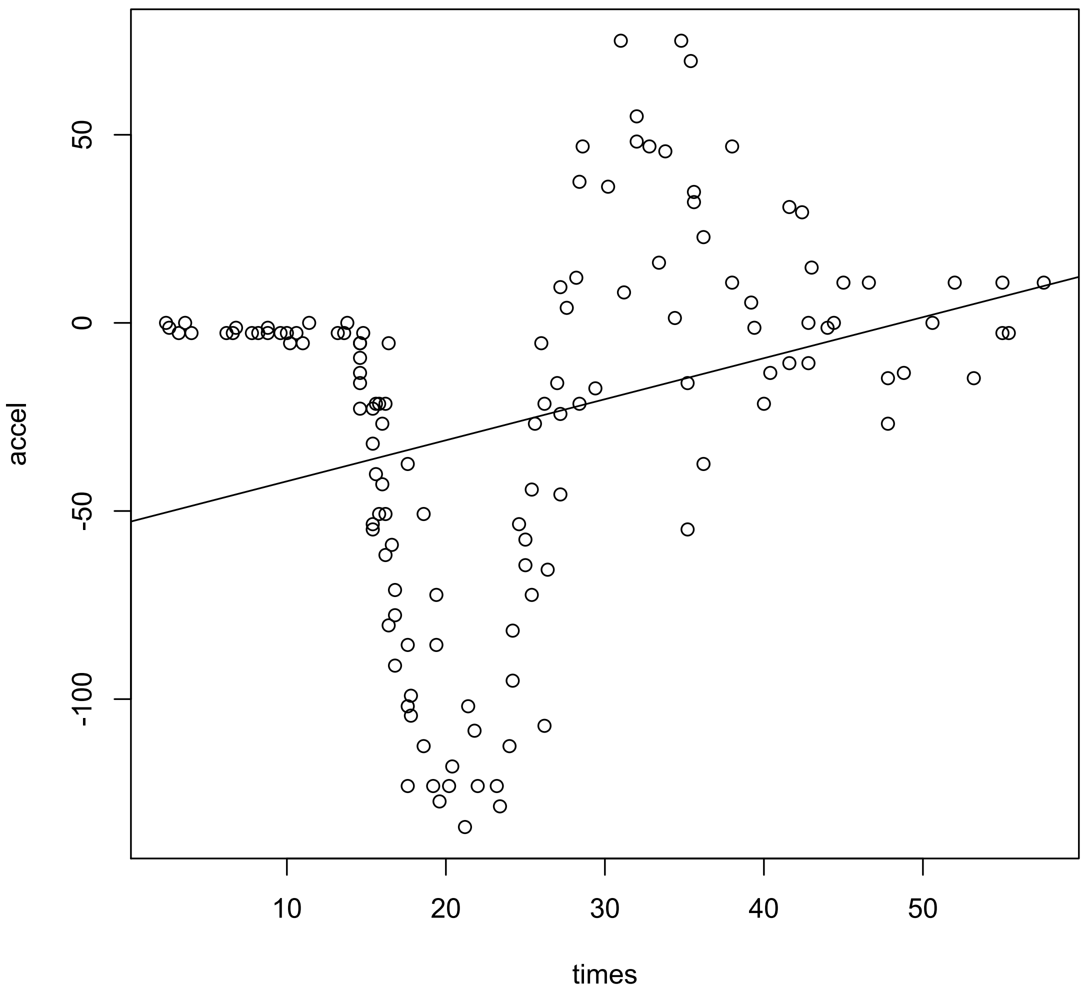
Motorcycle data

Head acceleration (in g) is measured against milliseconds after impact in a simulated motorcycle accident, used to test crash helmets.



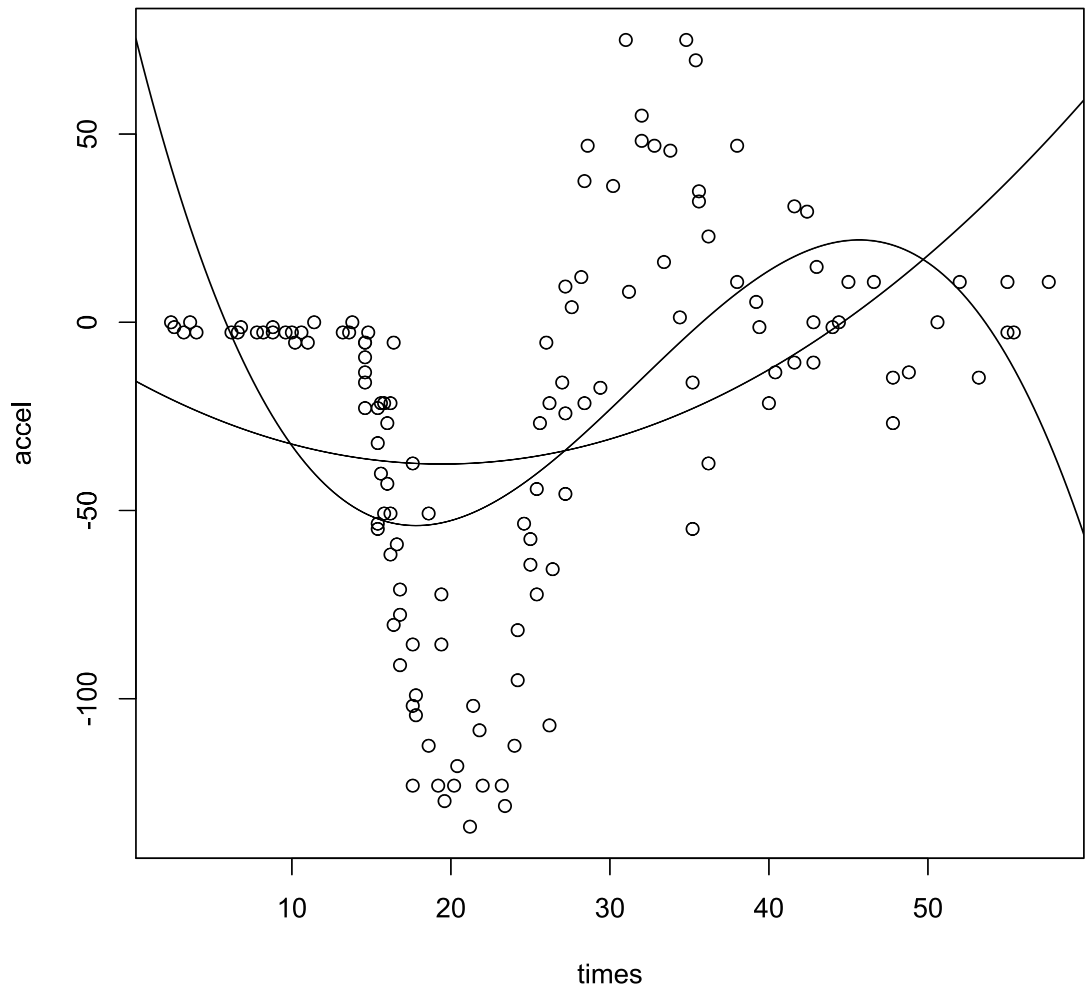
What is a good fit here?

Linear?



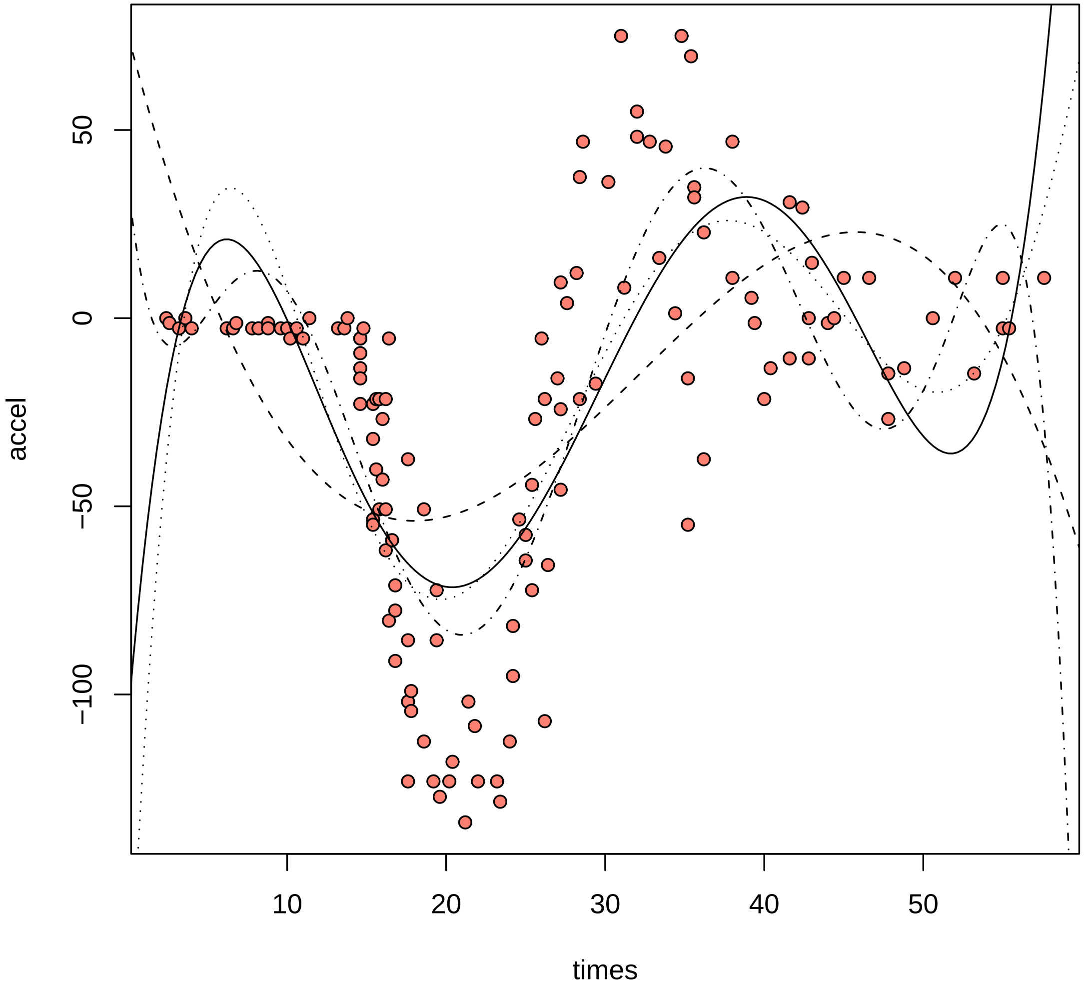

```
> plot(mcycle)
> abline(lm(accel~times,data=mcycle))
```

Quadratic? Cubic?



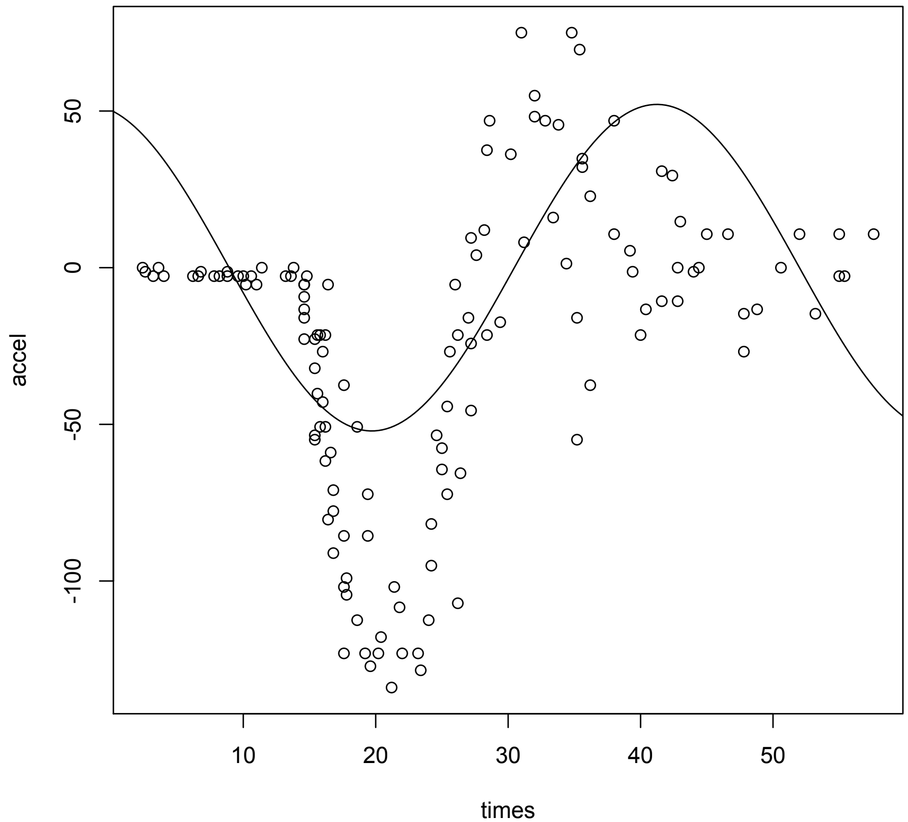

```
> plot(mcycle)
> lines(mcycx,predict(lm(accel~poly(times,2),data=mcycle),
+ data.frame(times=mcycx)))
> lines(mcycx,predict(lm(accel~poly(times,3),data=mcycle),
+ data.frame(times=mcycx)),lty=2)
```

Polynomial?



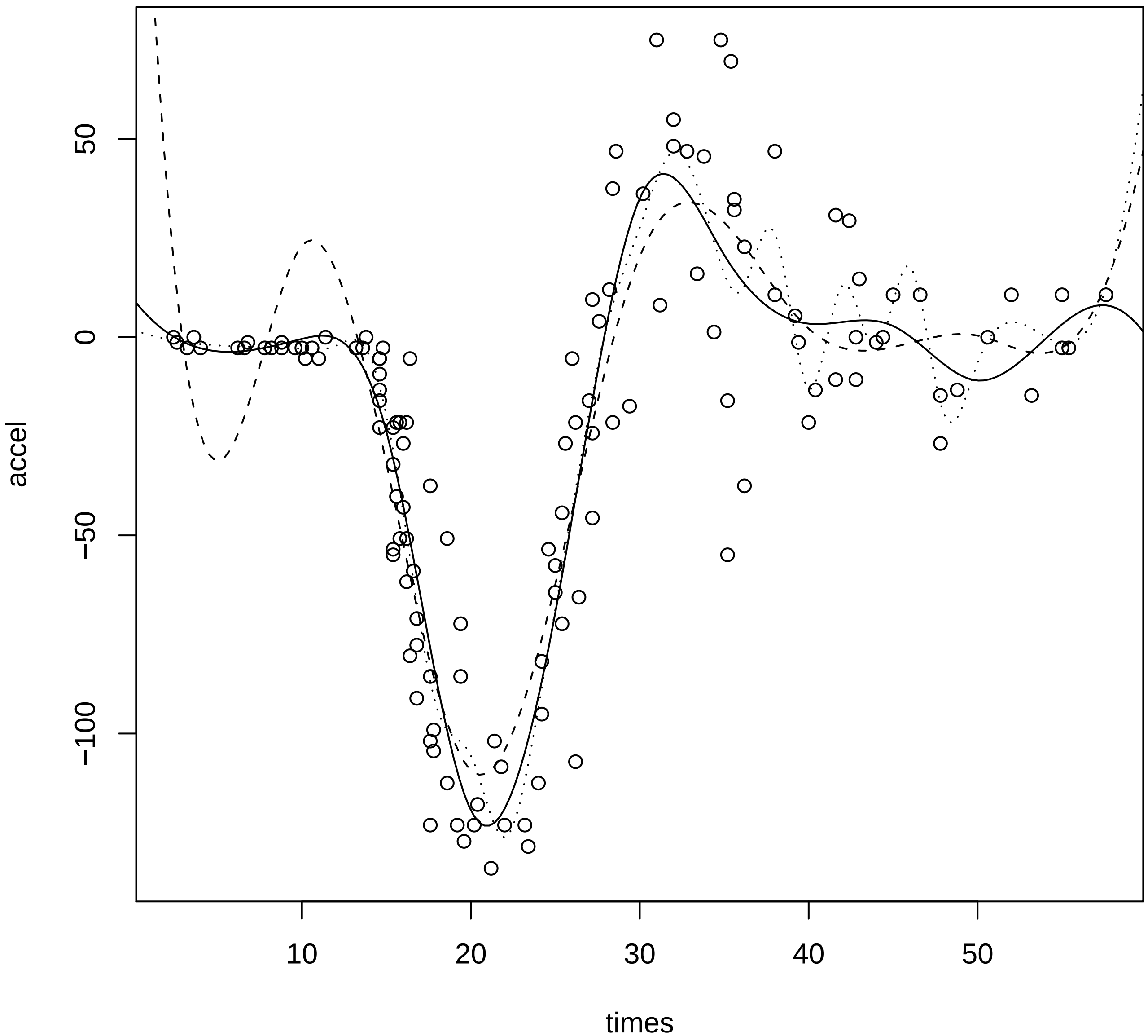

```
> lines(mcycx,predict(lm(accel~poly(times,4)),data.frame(times=mcycx)),lty=2)
> lines(mcycx,predict(lm(accel~poly(times,5)),data.frame(times=mcycx)),lty=1)
> lines(mcycx,predict(lm(accel~poly(times,6)),data.frame(times=mcycx)),lty=3)
> lines(mcycx,predict(lm(accel~poly(times,7)),data.frame(times=mcycx)),lty=4)
```

Trigonometric??




```
> mctrig
Nonlinear regression model
  model:  accel ~ p1 * sin(times/p2 + p3)
   data:  parent.frame()
      p1        p2        p3
52.10898  6.85355 96.08556
 residual sum-of-squares:  206615.8
> lines(mcycx,predict(mctrig,data.frame(times=mcycx)))
```

A way out: piecewise polynomials!



In R

```
> library(splines)
> plot(mcycle)
> lines(mcycx,predict(lm(accel ~ bs(times,
+ knots=seq(10,50,10))), data.frame(times=mcycx)),lty=2)
> lines(mcycx,predict(lm(accel ~ bs(times,
+ knots=seq(10,50,5))), data.frame(times=mcycx)))
> lines(mcycx,predict(lm(accel ~ bs(times,
+ knots=seq(10,50,2))), data.frame(times=mcycx)),lty=3)
```

Piecewise polynomials

Polynomial: $a_0 + a_1 x + a_2 x^2 + \dots a_p x^p$ (p degree)

Note: we don't require $a_p \neq 0$, so degree p contains all lower degrees; in particular, degree 0 polynomials are constants

Each polynomial is of class C^q for *every* q: it has q continuous derivatives (they are $= 0$ starting from $(p+1)$-th derivative)
Class C^0 contains all continuous functions

A polynomial of degree p is determined by
- the values at $p+1$ distinct points;
- or by the value at one point and the values of p derivatives;
- generally (but slightly imprecisely): by $p+1$ quantities

Polynomials with degree p form a linear space

Piecewise polynomial: function that is a polynomial on each of $(-\infty, \varkappa_1)$, $(\varkappa_1, \varkappa_2)$, ..., $(\varkappa_{K-1}, \varkappa_K)$, $(\varkappa_K, \infty)$

Of course, any polynomial is piecewise polynomial; we can consider the polynomials at all pieces to be of the same (maximal) degree p - *piecewise polynomial of degree* p

Points $\varkappa_k$ are customarily called **knots**; piecewise polynomials with fixed degree p and *fixed knots* form a]*linear space*

Linear spaces

When we specify, on the regressor side of an R formula, something like

```
poly(variable)    bs(variable)    ...
```

we are in fact fitting all the coefficients for a linear combination of certain *basis functions*, in a variable `variable`

Recall: *basis* - linearly independent collection of vectors spanning the linear space; here the linear space is the space of certain functions, functions that are linear combinations of the basis functions

In the mathematical formulation, if we denote the variable by x, we are fitting

$$y \sim \sum_{j=1}^{q} \beta_j g_j(x) + \text{possibly other regressors}$$

Nice flexible systems (unlike those coming from neural networks) can be written like this - so that the fitting procedure remains linear - that is, linear in β_j, not necessarily in x

A prominent example of such systems are *splines*

Special piecewise polynomials: splines

shipbuilding, automobile, airplane industry
mathematics; computer graphics

Spline: function piecewise polynomial of degree p and in C^{p-1}
- that is, continuous and have $p-1$ continuous derivatives
Note: in general, splines are *not* C^p

Examples of splines:
$p = 0$: piecewise constant functions
$p = 1$, linear splines: piecewise linear, continuous
$p = 2$, quadratic splines: continuous, continuous derivative
$p = 3$, cubic splines: continuous up to 2nd derivative

and also polynomials, as a special instance

Some piecewise polynomials are not splines:

cubic Bézier paths (more precisely, they coordinate components) typically require only C^1, not C^2 continuity

But, such things are also called "generalized splines" or "splines, given smoothness structure"; all those can be put under our spline definition, if multiple knots at the same point are allowed

Bases for polynomials

For polynomials, we often intuitively use the *monomial* basis:

$1, x, x^2, \ldots, x^q$ for example in R: `y ~ x + I(x^2) +I(x^3)`

This is not an *orthogonal* basis (a basis does not have to be necessarily so!), as we usually we do not have

$\int g_1(x) g_2(x)\, dx = 0$ for two different basis functions

If we write in R instead of the above `y ~ poly(x,3)`

we are using an orthogonal basis; examples of those are

Bernstein polynomials (restricted to $[0, 1]$ used in Bézier curves):

$\binom{n}{k} x^k (1 - x)^{n-k}$ $k = 0, 1, \ldots, n.$

However, in the R call of `poly(x,3)`, we actually use a basis orthogonal with respect to given points $x_1, \ldots, x_n$:

$\sum_i g_1(x_i) g_2(x_i) = 0$ for two different basis functions

Bases for splines

B-splines, Bézier splines, Hermite splines... do not mean different systems of functions, only different bases

That is, the fit will be the same (given the knots and the degree)

Some bases derive from those of polynomials, but there are also specific ones

An intuitive one is *truncated power* basis of degree p:

$$1, x, x^2, \ldots, x^p, (x - \varkappa_1)_+^p, (x - \varkappa_2)_+^p, \ldots, (x - \varkappa_m)_+^p$$

where $u_+ = u$ if $u \geqslant 0$, and $u_+ = 0$ if $u \leqslant 0$.

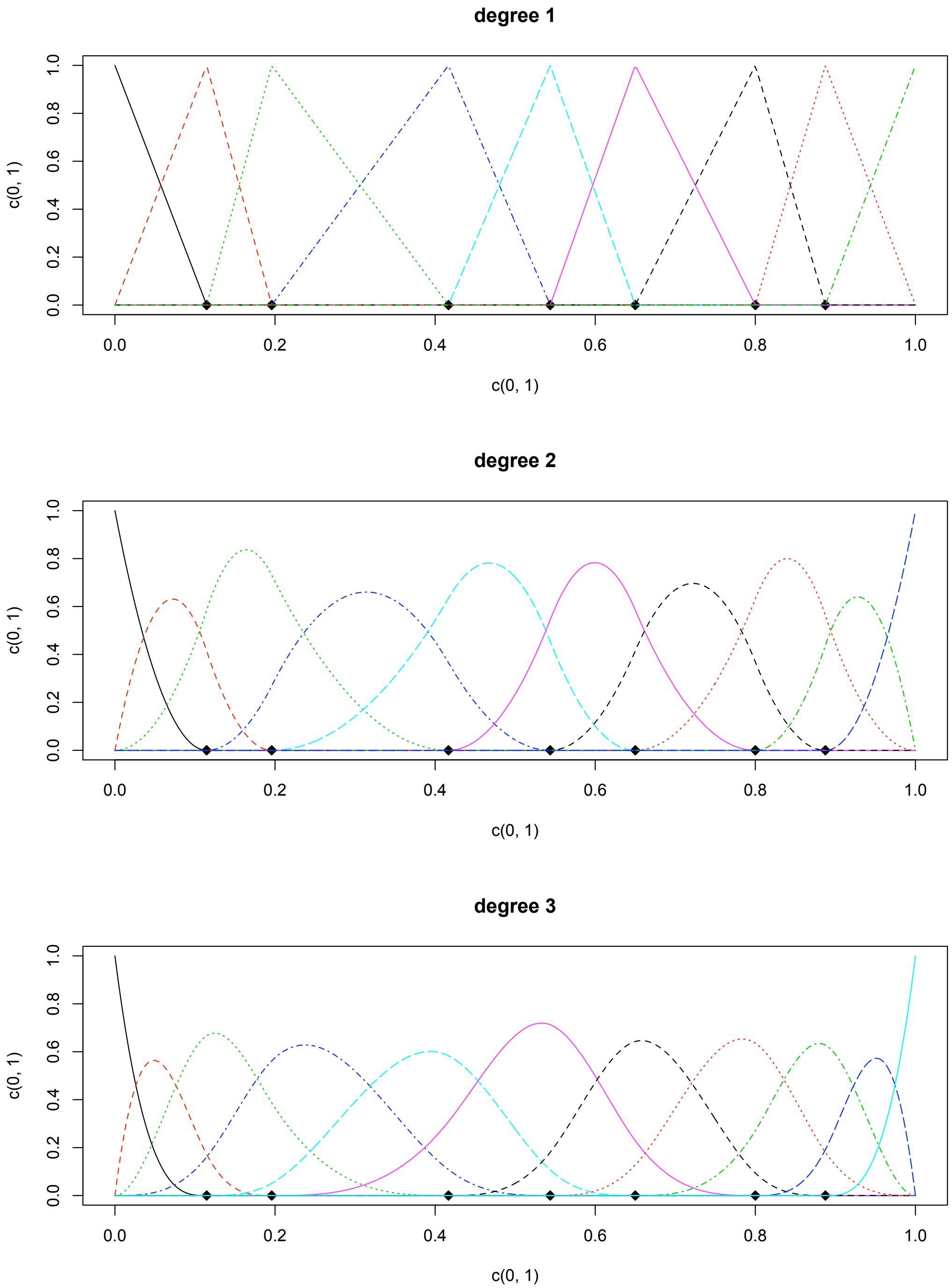
A similar one is *radial (thin-plate)* basis (for odd p)

$$1, x, x^2, \ldots, x^p, |x - \varkappa_1|^p, |x - \varkappa_2|^p, \ldots, |x - \varkappa_m|^p$$

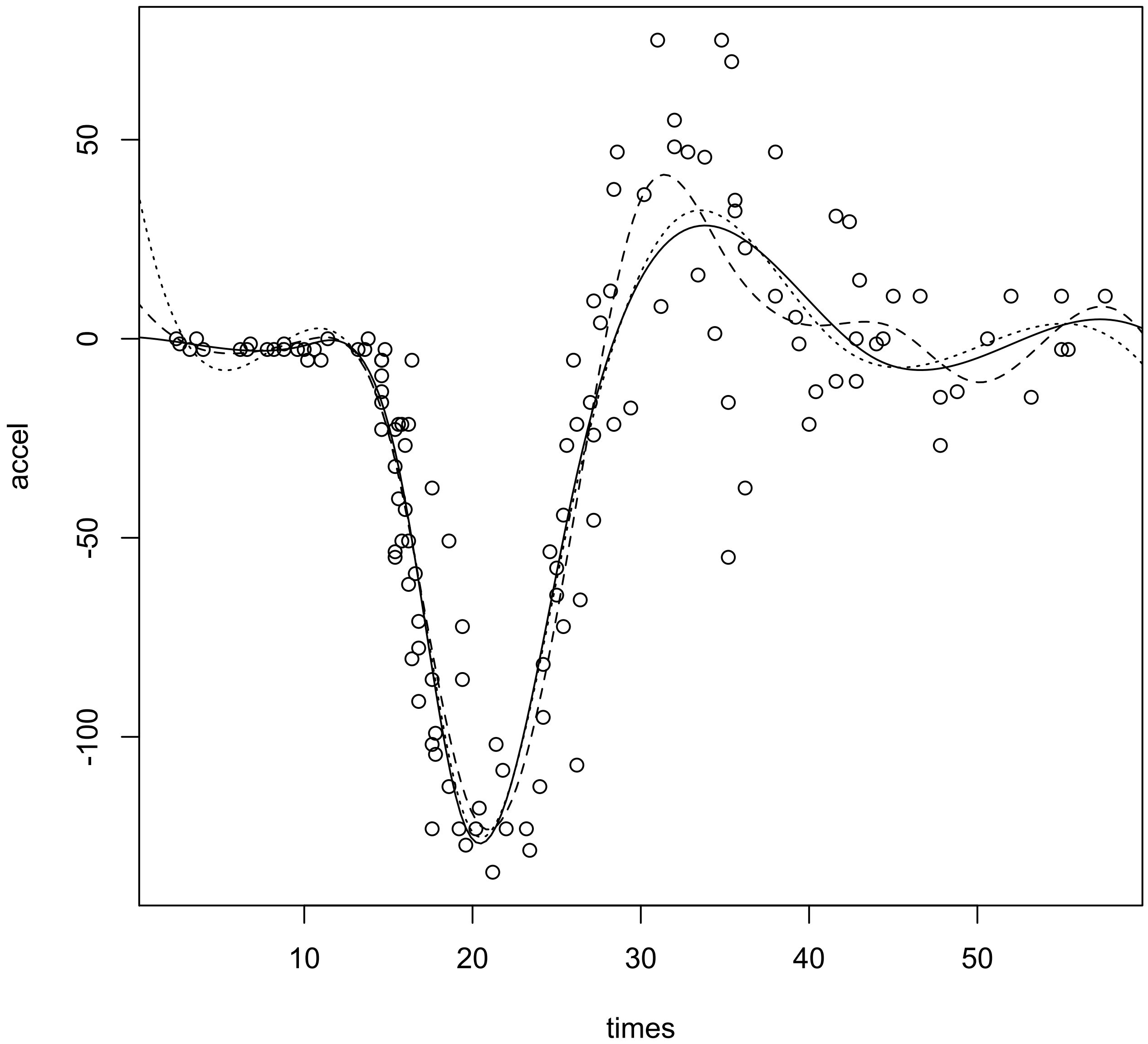
(generalizes well to higher dimensions)

Finally, the widely used one is *B-spline* basis, because of its several nice properties (which are, however, beyond the scope here)

Examples of B-splines



A fit of motorcycle data



Regression splines

```
> plot(mcycle)
> lines(mcycx,predict(lm(accel~bs(times,knots=seq(10,50,5))),
+ list(times=mcycx)),lty=2)
> lines(mcycx,predict(lm(accel~bs(times,knots=quantile(times,(1:6)/7))),
+ list(times=mcycx)),lty=3)
> lines(mcycx,predict(lm(accel~bs(times,knots=quantile(times,ppoints(6)))),
+ list(times=mcycx)))
```

So, what we had to specify here?

Degree - but we very often go for the *cubic splines* (degree 3)

Knots - the points where one polynomial changes to another

... and this may be a problem

If we are fitting like this - select degree, select knots, and fit - it is a usual linear regression (or generalized linear, in classification via logistic regression), which does not pose any numerical (and neither conceptual) problems

... we say, in this case, that we fit *regression splines*

And regression splines often fit well - if only we can handle:

The selection of knots

First problem: how many; second problem: where to place them

Regarding the first problem

Too few of them: the fit is not flexible

Too many of them: may overfit

We like the solutions that fit well - but we do not like those that are too wiggly.

Beware: making a knot a free parameter in a fitted model, results is a nonlinear problem (and a hard one, from several aspects)

Model selection strategies: focus thus rather on adding and removing knots using criteria like AIC, SIC, etc

The selection of knots: conclusion

Once we know how many:

- we may just choose them equispaced

- or, a common recommendation is to select knots at the quantiles of the covariate (see the code on the previous transparency)

- we may take equispaced quantiles, for probabilities $1/n$,$2/n$,..., $(n-1)/n$, or use `ppoints`, the function used in `qqnorm`

`ppoints()` default: $a = 3/8$ if $n \leqslant 10$, else $a = 1/2$

$$\frac{1-a}{n+(1-a)-a}, \ldots, \frac{n-a}{n+(1-a)-a}$$