Local regression II

Patrick Breheny

November 8

The loess function

- In R, local linear regression is implemented through the loess function, which uses a formula interface similar to that of other regression functions:

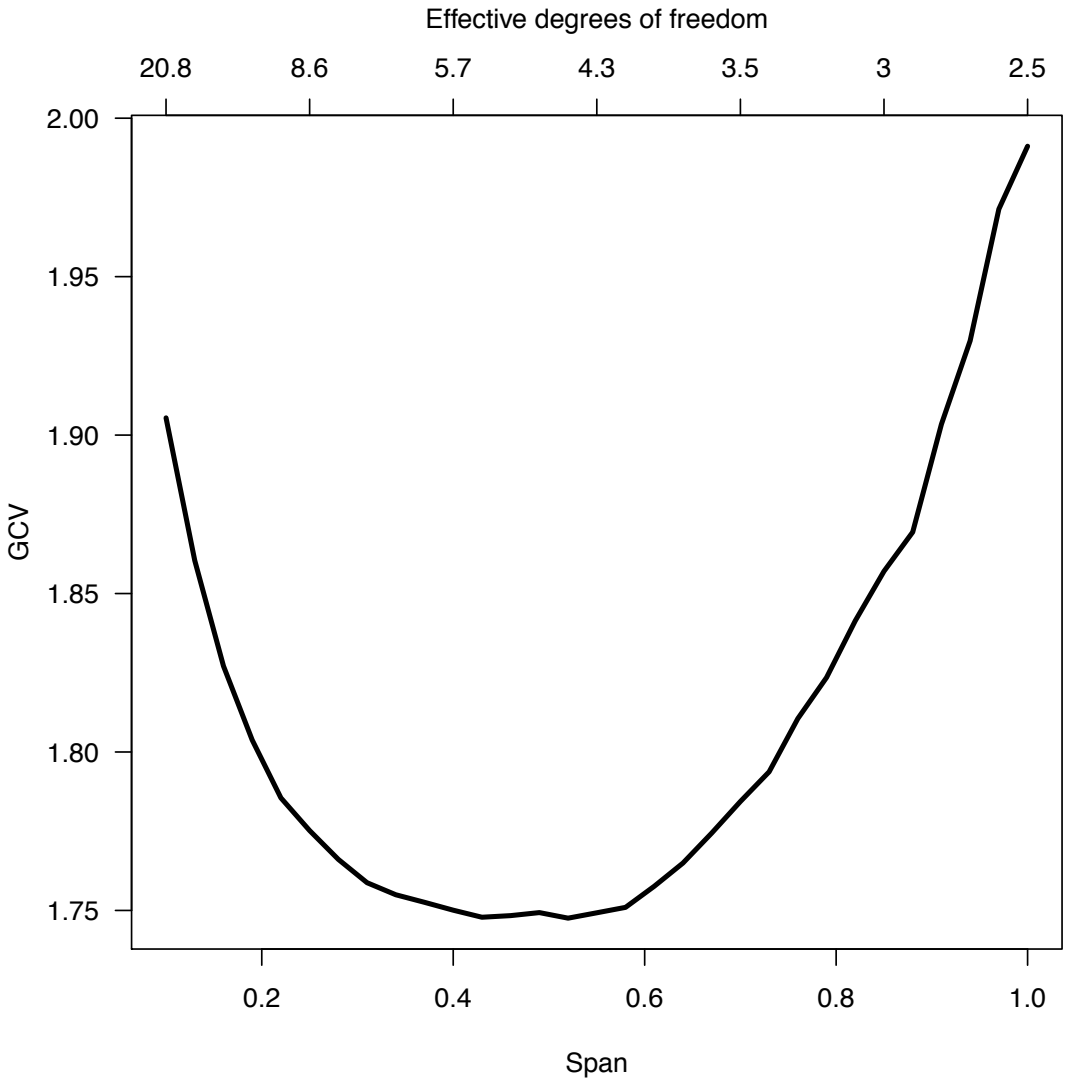
```
fit <- loess(spnbmd~age, m, span=0.3, degree=1)
```

- The two key options are
  - span: this is the smoothing parameter which controls the bias-variance tradeoff
  - degree: this lets you specify local constant regression (*i.e.*, the Nadaraya-Watson estimator, degree=0), local linear regression (degree=1), or local polynomial fits (degree=2, the default)

The span argument

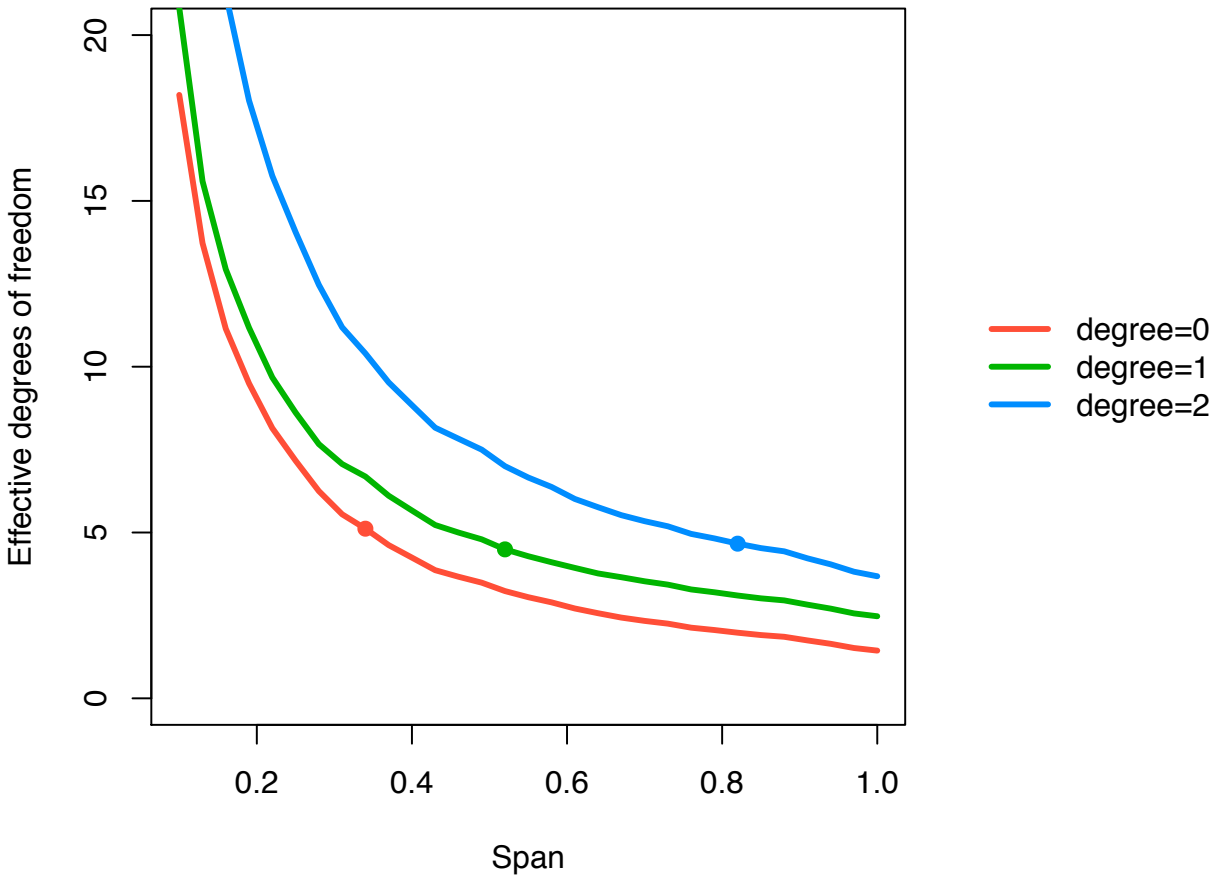
- Unlike `density`, `loess` does not allow you to choose your own kernel; only the tricube kernel is implemented, and `span` refers to the proportion of the observations $\{x_i\}$ within its compact support
- Also unlike `density`, the kernel in `loess` is adaptive
- Thus, specifying `span=0.2` means that the bandwidth of the kernel at x_0 is made just wide enough to include 20% of the x_i values
- The default value is 0.75, but this is just an ad-hoc suggestion; by no means is this always a good choice for the smoothing parameter

Selection of smoothing parameter

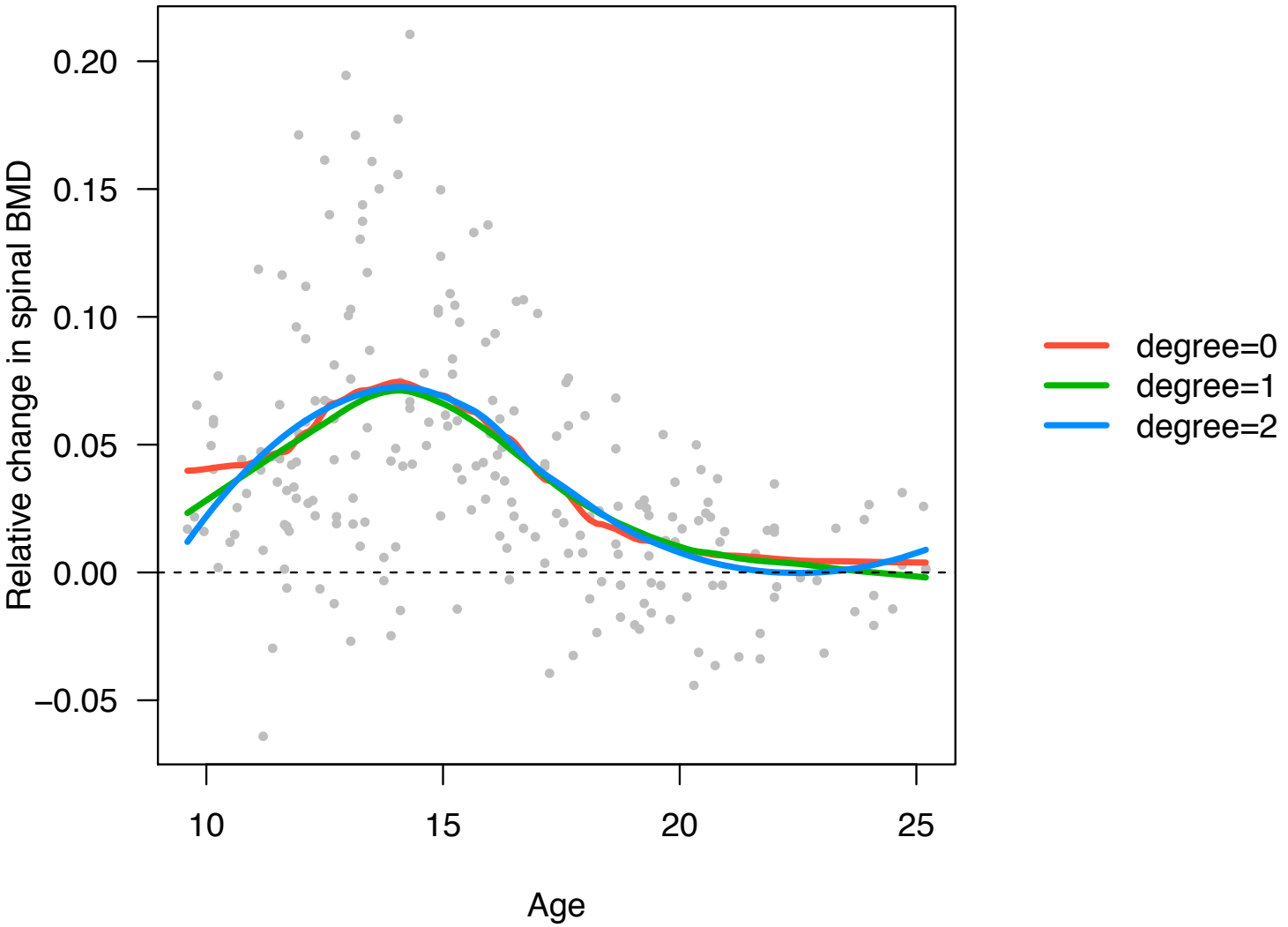


Effective degrees of freedom versus span

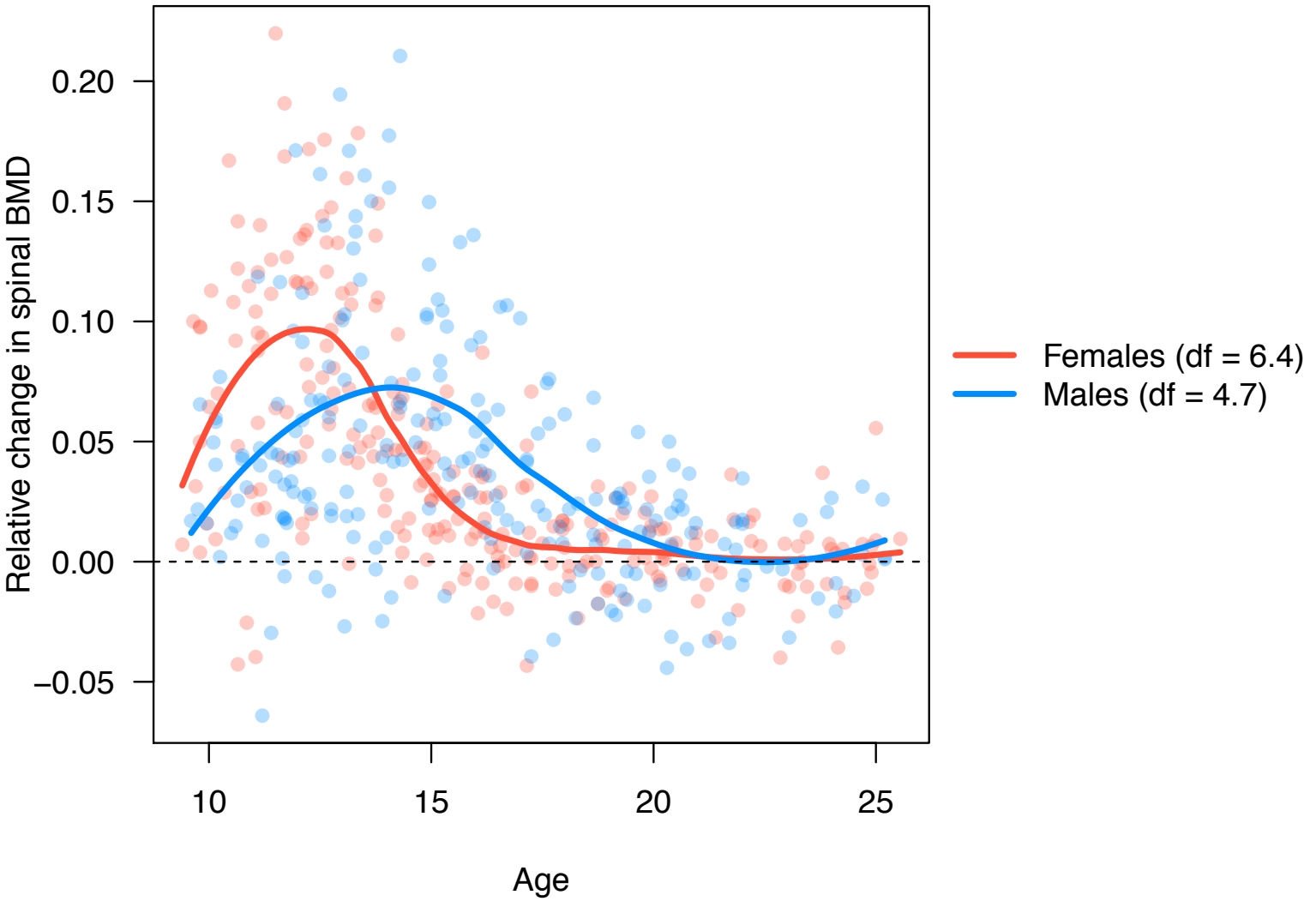
Dots indicate optimal smoothing, as chosen by GCV:



Optimal fits for the bone mineral density data



Bone mineral density data – males versus females



Other implementations: gam and locfit

- The `loess` function provides a convenient interface to fitting local polynomials, but carries out no inference (confidence bands, hypothesis tests, etc.)
- Inferential results are provided by the `gam` and `locfit` packages
- Each package has a somewhat different perspective and provides different tools, so they are both worth knowing about

Estimation of σ^2

- Note that

$$\mathbb{E}\sum_i (y_i - \hat{f}_i)^2 = \sigma^2(n - 2\nu + \tilde{\nu}) + \mathbf{b}'\mathbf{b},$$

where

$$\begin{aligned}\mathbf{b} &= \mathbb{E}(\mathbf{y}) - \mathbb{E}(\hat{\mathbf{f}}),\\ \nu &= \mathrm{tr}(\mathbf{L}),\\ \tilde{\nu} &= \mathrm{tr}(\mathbf{L}'\mathbf{L})\end{aligned}$$

- The bias term presents a problem, as it depends on the true regression function $\mathbf{f}$

Estimation of σ^2 (cont'd)

- However, if n is reasonably large compared with ν and $\tilde{\nu}$ and the bandwidth reasonably small, the effect of the bias term is negligible, and the following is a nearly unbiased estimator for σ^2:

$$\hat{\sigma}^2 = \frac{\sum_i (y_i - \hat{f}_i)^2}{n - 2\nu + \tilde{\nu}}$$

- The quantity $2\nu - \tilde{\nu}$ is known as the *equivalent number of parameters*, by analogy with linear regression

The `locfit` function

- This estimate, along with ν and $\tilde{\nu}$, are provided by the `locfit` package
- The basic syntax of model fitting with `locfit` is as follows:

  ```
  fit <- locfit(spnbmd~lp(age, nn=.7, deg=2))
  ```

  where `lp` controls the local polynomial which is fit to the data
- Just like `loess`, there is a `nn` parameter (analogous to `span`), which adaptively determines the bandwidth by setting the number of points in the neighborhood of x_0 equal to `nn`
- There is also a `deg` parameter, which controls the degree of the local polynomial (like `loess`, the default is 2)

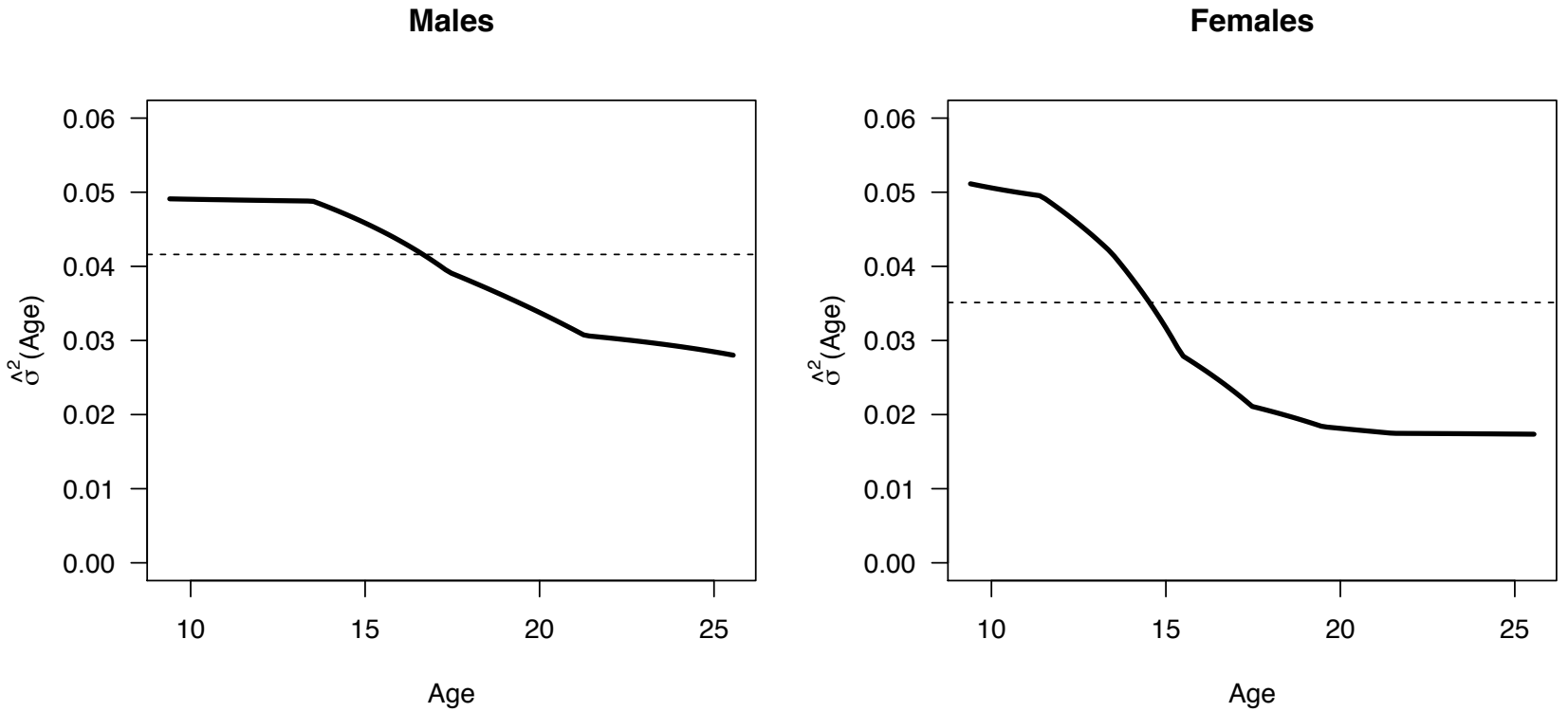
Obtaining $\hat{\sigma}^2$ from `locfit`

- The fitted object returned by `locfit` contains a (rather poorly documented) component called `dp` which contains information about the fit:
  - `df1`: ν
  - `df2`: $\tilde{\nu}$
  - `lk`: the log-likelihood (actually, $-\mathrm{RSS}/2$)
  - `rv`: $\hat{\sigma}^2$
- Note that `-2*lk/(n-2*df1+df2)` equals `rv`
- For the BMD data, $\hat{\sigma} = 0.42$ for the males and $\hat{\sigma} = 0.35$ for the females

Estimation of σ^2 (cont'd)

- The preceding estimate assumes homoskedasticity (constant variance)
- An alternative, given that we are fitting local linear models, is to estimate $\hat{\sigma}^2(x_0)$ from the usual linear regression estimate for the model fitted at x_0
- The information needed to calculate these estimates is available with `predict.locfit` (see the accompanying code for details)

Local variance estimates



Pointwise inference

- Recall that

$$\mathbb{E}\hat{f}(x_0) = \sum_i l_i(x_0) f(x_0)$$
$$\mathbb{V}\hat{f}(x_0) = \sigma^2 \|l(x)\|^2,$$

- One method of constructing pointwise confidence intervals, then, is via

$$\hat{f}(x_0) \pm z_{\alpha/2}\hat{\sigma} \|l(x_0)\|,$$

The bias problem

- However, as we have remarked several times, $\hat{f}$ is not an unbiased estimate of f:

$$\frac{\hat{f} - f}{\text{SE}} \sim Z + \frac{\text{bias}}{\text{SE}},$$

  where $Z \sim N(0, 1)$
- Thus, usual normal-theory methods for confidence intervals will result in confidence intervals for $\bar{f} = \mathbb{E}(\hat{f})$, not for f itself
- Generally, however, the bias term is just ignored and we just accept the fact that the interval is technically an interval for $\bar{f}$, not f

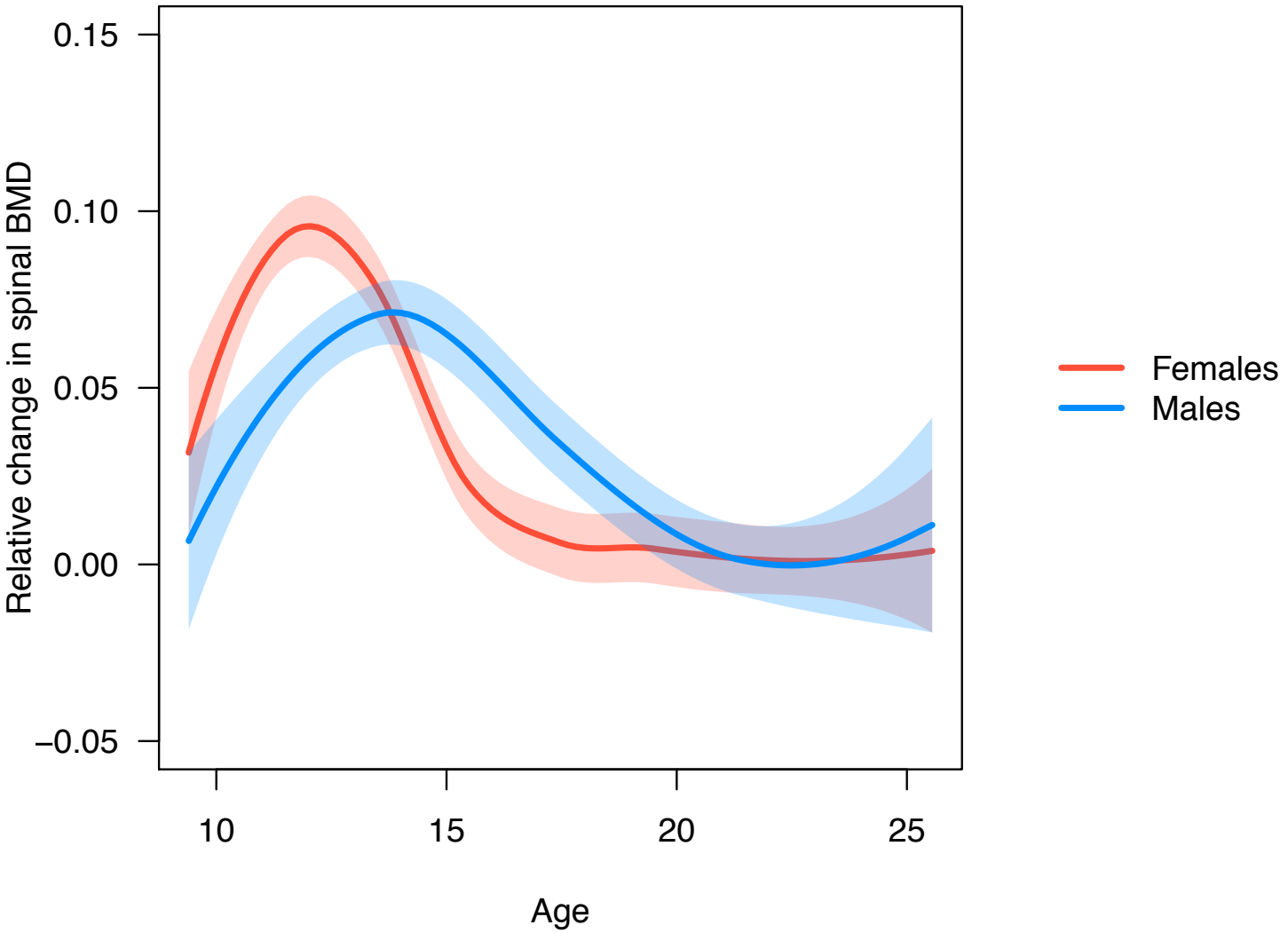
Global confidence bands

- It is also possible to obtain simultaneous confidence bands across the entire range of x
- The details are fairly complicated and rest on representing

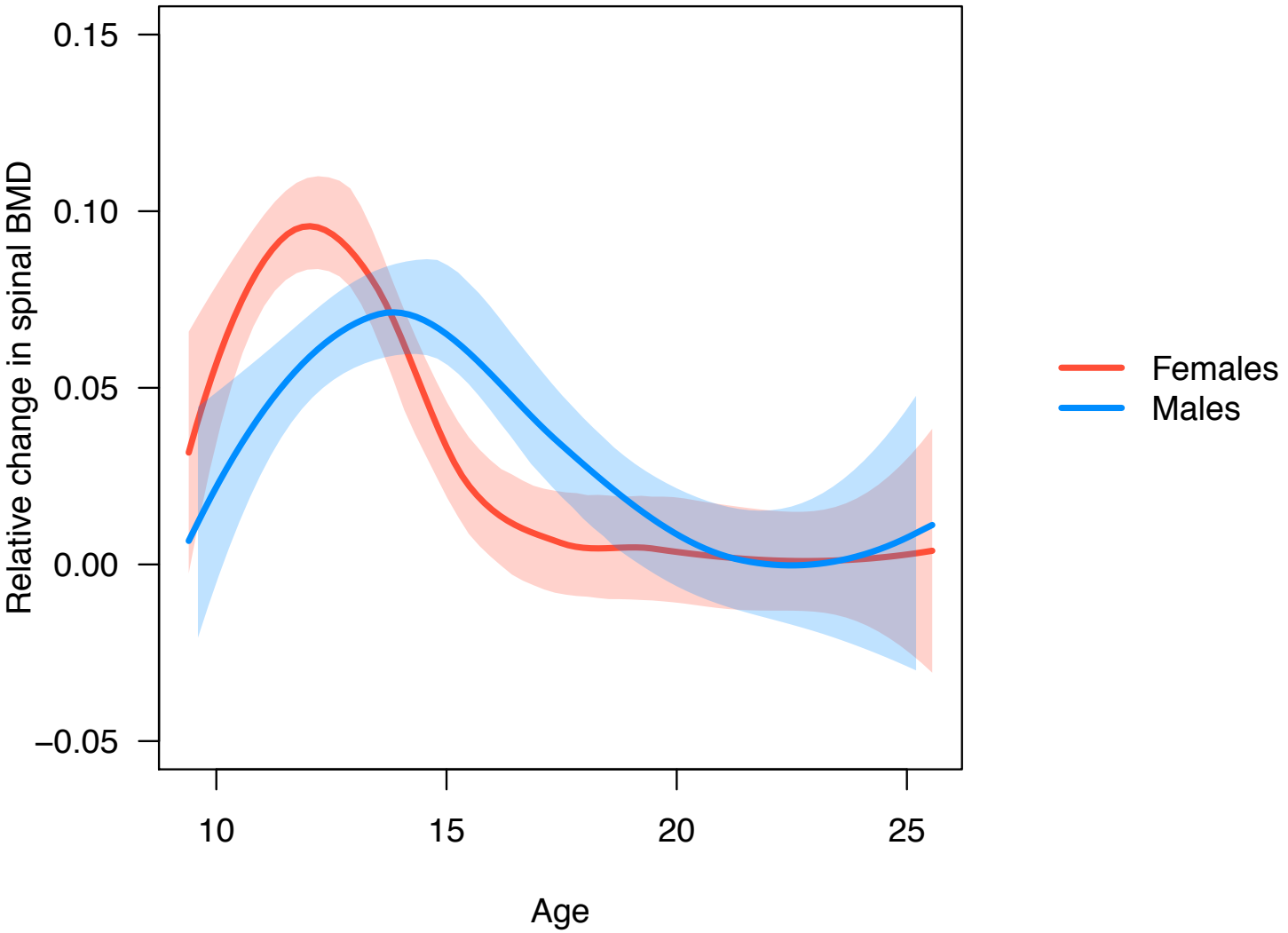
$$W(x) = \frac{\hat{f}(x) - \bar{f}(x)}{\sigma \, \|l(x)\|}$$

  as a Gaussian process
- We won't go into the details (they are in Section 5.7 of our text), but we will mention that the `scb` function in `locfit` computes simultaneous confidence bands (again, details are in the accompanying code)

Pointwise CIs for the bone mineral density data



Simultaneous CIs for the bone mineral density data



Hypothesis testing

- Lastly, we consider the issue of hypothesis testing
- Consider the following nested sequence of models:

$$\text{M0: } \mathbb{E}(Y|x) = \alpha$$
$$\text{M1: } \mathbb{E}(Y|x) = \alpha + \beta x$$
$$\text{M2: } \mathbb{E}(Y|x) = f(x)$$

- We may be interested in testing null hypotheses such as whether there is any relationship between x and y, or whether a linear parameterization of the relationship provides an adequate fit

F tests

- The idea of the F test from conventional linear models applies here as well:

$$F = \frac{(\mathrm{RSS}_0 - \mathrm{RSS}_1)/(\mathrm{df}_1 - \mathrm{df}_0)}{\mathrm{RSS}_1/(n - \mathrm{df}_1)},$$

  where $\mathrm{df} = 2\nu - \tilde{\nu}$

- Recall that there is a bias term present that we are ignoring; thus F follows an $F_{\mathrm{df}_1 - \mathrm{df}_0, n - \mathrm{df}_1}$ distribution only asymptotically

The gam function

- One can obviously calculate this quantity by hand using the `locfit` package, but it is usually more convenient to use the `gam` package
- The basic syntax of model fitting with `gam` is very similar to `loess` and `locfit`:

```
fit <- gam(spnbmd~lo(age, span=.5, deg=1))
```

where `lo` controls the local polynomial which is fit to the data

The anova function

One can then compare a sequence of nested models using `anova.gam`:

```
fit0 <- gam(spnbmd~1, data=m)
fit1 <- gam(spnbmd~age, data=m)
fit2 <- gam(spnbmd~lo(age), data=m)
anova(fit0, fit1, fit2, test="F")
```

	Rdf	RSS	Δdf	ΔRSS	F	$\mathbb{P}(>F)$
M0	225	0.53				
M1	224	0.45	1	0.08	47.91	< 0.0001
M2	221	0.38	3	0.07	16.25	< 0.0001