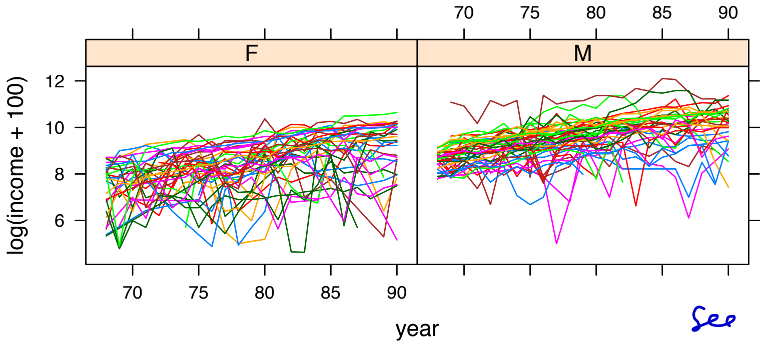
Today

- data presentation Yi Lu
- re-cap on random effects examples
- in the news
- semi-parametric regression
- March/April: Semi-parametric regression (§10.7), generalized additive models, penalized regression methods (ridge regression, lasso); proportional hazards models (§10.8)
- Chapter 9 reading: 9.1, 9.2.1, 9.2.2, 9.3.1, 9.3.2, 9.4
- HW 3: due March 21

Example: Panel Study of Income Dynamics Faraway, §9.1



see rat.
growth

$$\log(\text{income})_{ij} = \mu + b_{0j} + \alpha\,\text{year}_i + b_{1j}\text{year}_i + \beta\,\text{sex}_j + \gamma\,(\text{year}_i \times \text{sex}_j) + \beta_2\text{educ}_j + \beta_3\text{age}_j + \epsilon_{ij},$$

$$\epsilon_{ij} \sim N(0, \sigma^2), \quad b_j \sim N_2(0, \sigma^2\Omega_b)$$

year = year − 78 j subject, i year

... PSID

```
> mmod = lmer(log(income) ~ cyear*sex + age + educ +
+ (cyear | person), data=psid)
```

$$
\begin{aligned}
\log(\text{income})_{ij} \quad = \quad & \mu + b_{0j} + \alpha\,\text{year}_i + b_{1j}\text{year}_i + \\
& \beta\,\text{sex}_j + \gamma\,(\text{year}_i \times \text{sex}_j) + \beta_2\text{educ}_j + \beta_3\text{age}_j + \epsilon_{ij}, \\
& \epsilon_{ij} \sim N(0, \sigma^2), \quad b_j \sim N_2(0, \sigma^2\Omega_b)
\end{aligned}
$$

- we could fit separate lines for each subject (as also mentioned in SM Example 9.18)
- this would give us 85 slopes and 85 intercepts
- we could compare these slopes and intercepts between genders (two-sample test)
- simple, but limited

... PSID – using `lmer`

compare random effects model to fixed effects model:

$[\varepsilon \sim N(0, \Omega)]$

$\varepsilon \sim N(0, \sigma^2 I)$

```
> mmod = lmer(log(income) ~ cyear*sex + age + educ +
+ (cyear | person), data=psid)

Fixed effects:
             Estimate Std. Error t value
(Intercept)  6.67420    0.54332  12.284
cyear        0.08531    0.00900   9.480
sexM         1.15031    0.12129   9.484
age          0.01093    0.01352   0.808
educ         0.10421    0.02144   4.861
cyear:sexM  -0.02631    0.01224  -2.150
```

$y = X\beta + Zb + \varepsilon$

$\int db \qquad y \sim N(X\beta, \sigma^2 \Gamma^{-1})$

unclear

$\Omega_b \qquad \sigma^2$

```
> lmod = lm(log(income) ~ cyear*sex + age + educ, data = paid)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.737201   0.206490  32.627   <2e-16 ***
cyear        0.082049   0.005304  15.470   <2e-16 ***
sexM         1.130826   0.045554  24.824   <2e-16 ***
age          0.009401   0.005061   1.858   0.0634 .
educ         0.106934   0.008184  13.066   <2e-16 ***
cyear:sexM  -0.017716   0.007088  -2.499   0.0125 *

Residual standard error: 0.9126 on 1655 degrees of freedom
```

- coefficients the same; standard errors for `lm` much smaller
- 1655 degrees of freedom?
- all observations treated as independent

... PSID – using `lme`

$1661 - 2 = 1659$

```
> mmod = lme(log(income) ~ cyear*sex + age + educ ,
random = ~ 1 +  cyear | person, data=psid)
```

fixed effects

$1659 - 85$ for subject

fit

1574

```
Fixed effects: log(income) ~ cyear * sex + age + educ
                 Value Std.Error   DF   t-value p-value
(Intercept)   6.674204 0.5433252 1574 12.283995  0.0000
cyear         0.085312 0.0089996 1574  9.479521  0.0000
sexM          1.150313 0.1212925   81  9.483790  0.0000
age           0.010932 0.0135238   81  0.808342  0.4213
educ          0.104210 0.0214366   81  4.861287  0.0000
cyear:sexM   -0.026307 0.0122378 1574 -2.149607  0.0317
```

one random

```
Random effects:
 Formula: ~1 + cyear | person
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev     Corr
(Intercept) 0.53071321 (Intr)
cyear       0.04898952 0.187
Residual    0.68357323
```

σ_{b_0} $\rho(b_0, b_1)$ ~~$\sigma_{b_0}\sigma_{b_1}$~~

$\epsilon \sim (0, \sigma^2)$ $\hat{\sigma}$ σ_{b_1}

```
> lmod = lm(log(income) ~ cyear*sex + age + educ, data = paid)
Coefficients:
             Estimate  Std. Error t value Pr(>|t|)
(Intercept)  6.737201   0.206490  32.627   <2e-16 ***
cyear        0.082049   0.005304  15.470   <2e-16 ***
sexM         1.130826   0.045554  24.824   <2e-16 ***
age          0.009401   0.005061   1.858   0.0634 .
educ         0.106934   0.008184  13.066   <2e-16 ***
cyear:sexM  -0.017716   0.007088  -2.499   0.0125 *

Residual standard error: 0.9126 on 1655 degrees of freedom
```

!

Inference for fixed effects

$y \sim N(X\beta, \sigma^2 \Upsilon^{-1})$

- $\hat\beta = (X^{\mathrm{T}}\hat\Upsilon X)^{-1}X^{\mathrm{T}}\hat\Upsilon y, \quad \hat\sigma^2 = \frac{1}{n}(y - X\hat\beta)^{\mathrm{T}}(y - X\hat\beta)$
- $\hat\sigma^2$ usually replaced by REML estimate $\tilde\sigma^2$
- s.e.$(\hat\beta_j) = \sqrt{\{\tilde\sigma^2(X^{\mathrm{T}}\hat\Upsilon X)^{-1}_{jj}\}}$
- `educ` coefficient estimate 0.1042, $e^{0.1042} = 1.11$, 11% increase in income per year of education
- `sexM` coefficient estimate 1.15, $e^{1.15} = 3.16$, $3\times$ higher at baseline for males
- slope for females approximately 9% per year; for males approximately 6% per year
- standard deviation of slopes estimated to be 0.049
- variation within subjects $(0.68)^2$ larger than between subjects $(0.53)^2$

Inference for fixed effects

- $\hat{\beta} = (X^{\mathrm{T}}\hat{\Upsilon}X)^{-1}X^{\mathrm{T}}\hat{\Upsilon}y, \quad \hat{\sigma}^2 = \frac{1}{n}(y - X\hat{\beta})^{\mathrm{T}}(y - X\hat{\beta})$
- $\hat{\sigma}^2$ usually replaced by REML estimate $\tilde{\sigma}^2$
- s.e.$(\hat{\beta}_j) = \surd\{\tilde{\sigma}^2(X^{\mathrm{T}}\hat{\Upsilon}X)^{-1}_{jj}\}$
- `educ` coefficient estimate 0.1042, $e^{0.1042} = 1.11$, 11% increase in income per year of education
- `sexM` coefficient estimate 1.15, $e^{1.15} = 3.16$, $3\times$ higher at baseline for males
- slope for females approximately 9% per year; for males approximately 6% per year
- standard deviation of slopes estimated to be 0.049
- variation within subjects $(0.68)^2$ larger than between subjects $(0.53)^2$

Random effects

- estimates (predictions) of b_{0i}, b_{1i} available

  j for subject

- $Y = X\beta + Zb + \epsilon; \quad b \sim N(0, \sigma^2\Omega_b), \epsilon \sim N(0, \sigma^2\Omega_j)$

- $Y \sim N(X\beta, (\Omega + Z\Omega_b Z^{\mathrm{T}}))$

- $\tilde{b} = (Z^{\mathrm{T}}\hat{\Omega}^{-1}Z + \hat{\Omega}_b^{-1})^{-1} Z^{\mathrm{T}}\Omega^{-1}(y - X\beta)$

$$
\begin{aligned}
y - X\hat{\beta} &= Z\tilde{b} + y - X\hat{\beta} - Z\tilde{b} \\
&= Z\tilde{b} + \underbrace{\{I_n - Z(Z^{\mathrm{T}}\hat{\Omega}^{-1}Z + \hat{\Omega}_b^{-1})^{-1}Z^{\mathrm{T}}\hat{\Omega}^{-1}\}(y - X\hat{\beta})}_{\text{new residual}}
\end{aligned}
$$

Random effects

- estimates (predictions) of b_{0i}, b_{1i} available
- $Y = X\beta + Zb + \epsilon; \quad b \sim N(0, \sigma^2\Omega_b), \epsilon \sim N(0, \sigma^2\Omega_j)$
- $Y \sim N(X\beta, (\Omega + Z\Omega_b Z^{\mathrm{T}}))$
- $\tilde{b} = (Z^{\mathrm{T}}\hat{\Omega}^{-1}Z + \hat{\Omega}_b^{-1})^{-1}Z^{\mathrm{T}}\hat{\Omega}^{-1}(y - X\hat{\beta})$

a bit like WLS
BLUP
estimate of E(b|y)

$$\begin{aligned} y - X\hat{\beta} &= Z\tilde{b} + y - X\hat{\beta} - Z\tilde{b} \\ &= Z\tilde{b} + \underbrace{\{I_n - Z(Z^{\mathrm{T}}\hat{\Omega}^{-1}Z + \hat{\Omega}_b^{-1})^{-1}Z^{\mathrm{T}}\hat{\Omega}^{-1}\}(y - X\hat{\beta})}_{\text{new residual}} \end{aligned}$$

Random effects

- estimates (predictions) of b_{0i}, b_{1i} available
- $Y = X\beta + Zb + \epsilon; \quad b \sim N(0, \sigma^2\Omega_b), \epsilon \sim N(0, \sigma^2\Omega_j)$
- $Y \sim N(X\beta, (\Omega + Z\Omega_b Z^{\mathrm{T}}))$
- $\tilde{b} = (Z^{\mathrm{T}}\hat{\Omega}^{-1}Z + \hat{\Omega}_b^{-1})^{-1}Z^{\mathrm{T}}\Omega^{-1}(y - X\beta)$

$$
\begin{aligned}
y - X\hat{\beta} &= Z\tilde{b} + y - X\hat{\beta} - Z\tilde{b} \\
&= Z\tilde{b} + \underbrace{\{I_n - Z(Z^{\mathrm{T}}\hat{\Omega}^{-1}Z + \hat{\Omega}_b^{-1})^{-1}Z^{\mathrm{T}}\hat{\Omega}^{-1}\}(y - X\hat{\beta})}_{\text{new residual}}
\end{aligned}
$$

Shrinkage

pieces of `lmer`

~~methods(mmod)~~
class(mmod)

```
> methods(class="merMod")
 [1] anova.merMod*         as.function.merMod*   coef.merMod*
 [4] confint.merMod        deviance.merMod*      drop1.merMod*
 [7] extractAIC.merMod*    family.merMod*        fitted.merMod*
[10] fixef.merMod*         formula.merMod*       isGLMM.merMod*
[13] isLMM.merMod*         isNLMM.merMod*        isREML.merMod*
[16] logLik.merMod*        model.frame.merMod*   model.matrix.merMod*
[19] nobs.merMod*          plot.merMod*          predict.merMod*
[22] print.merMod*         profile.merMod*       ranef.merMod*
[25] refit.merMod*         refitML.merMod*       residuals.merMod*
[28] sigma.merMod*         simulate.merMod*      summary.merMod*
[31] terms.merMod*         update.merMod*        VarCorr.merMod*
[34] vcov.merMod           weights.merMod*
```

$\tilde{b}_{0j}$ $\tilde{b}_{1j}$ BLUPS

```
> ranef(mmod)
$person
   (Intercept)        cyear
1  -0.029975590  0.0161575447
2   0.015961618  0.0198586106
3  -0.122972629 -0.0449473569
4   0.109534933 -0.0074016139
5  -0.572308284 -0.1108678330
6   0.218592408  0.0263156155
```

85 one/subj.

```
> length(residuals(mmod))
[1] 1661
```

Example: Balance experiment

Faraway, 10.1

- 3×2 factorial, 2 replications per subject
- factors: surface (normal or foam);
  vision (open, closed, domed)
- 20 male and 20 female subjects
- auxiliary variables age, height, weight

- simplest analysis, subject by subject 2×3 factorial with 2 observations per cell

	1	2	3
1	xx	xx	xx
2	xx	xx	xx

12

surface	(1)
vis	(2)
s × v	(2)
error	6

... balance

- three possible model fits
  1. ignore subject, fit usual `glm`
  2. include a fixed effect for each subject, fit usual `glm` – confounded with subject-level covariates
  3. include random intercepts for subject – fewer parameters to estimate, allows subject covariates to be used
- fit using `glmer` in `lme` or `glmmPQL` in `MASS`
- each involves an approximate integral of random effects, results can vary depending on control parameters

$$f(y_i \mid b) \sim \mathrm{Bin}(1, p_i) \qquad \mathrm{logit}\, p_i = x_i^T \beta + \cancel{z_i^T b} + b_i$$

$$f(y_i) = \int \underbrace{\qquad\qquad}_{} \cdot f(b)\, db$$

... balance

- three possible model fits
  1. ignore subject, fit usual `glm`
  2. include a fixed effect for each subject, fit usual `glm` – confounded with subject-level covariates
  3. include random intercepts for subject – fewer parameters to estimate, allows subject covariates to be used
- fit using `glmer` in `lme` or `glmmPQL` in `MASS`
- each involves an approximate integral of random effects, results can vary depending on control parameters

... balance

```
> library(MASS)

> balance2 <- glmmPQL(stable ~ Sex + Age + Height + Weight + Surface + Vision,
+ random = ~1 | Subject, family = binomial, data = ctsib)
> summary(balance2)

Random effects:
 Formula: ~1 | Subject
        (Intercept)  Residual
StdDev:    3.060712 0.5906232

Variance function:
 Structure: fixed weights
 Formula: ~invwt

Fixed effects: stable ~ Sex + Age + Height + Weight + Surface + Vision
                Value Std.Error  DF   t-value p-value
(Intercept) 15.571494 13.498304 437  1.153589  0.2493
Sexmale      3.355340  1.752614  35  1.914478  0.0638
Age         -0.006638  0.081959  35 -0.080992  0.9359
Height      -0.190819  0.092023  35 -2.073601  0.0455
Weight       0.069467  0.062857  35  1.105155  0.2766
Surfacenorm  7.724078  0.573578 437 13.466492  0.0000
Visiondome   0.726464  0.325933 437  2.228873  0.0263
Visionopen   6.485257  0.543980 437 11.921876  0.0000
```

$480 - 3 = 477 - 40 = 437 \qquad 40 - 5 = 35$

this is similar to a split-plot experiment: treatments are within subjects (sub-plots);

covariates are between subjects (main plots); see OzDASL

In the News

CBC


Cheating students punished by the 1000s, but many more go undetected - Manitoba - CBC News

www.cbc.ca/news/canada/manitoba/cheating-students-punished-by-the-1000s-but-many-more-go-undetected

CBCnews | Manitoba

LIVE Manitoba More

Radio One

Listen Live

Home | World | Canada | Politics | Business | Health | Arts & Entertainment | Technology & Science | Community | Weath

Canada › Manitoba › Photo Galleries

Cheating students punished by the 1000s, but many m undetected

CBC survey shows 7,086 students disciplined for cheating at Canadian universities in 2011-12

By Holly Moore, CBC News   Posted: Feb 25, 2014 4:00 AM CT | Last Updated: Feb 25, 2014 4:30 PM CT

Stay Connected with CBC Ne

Mobile   Facebook   Podcasts   Twitter

2014 TIGUAN

... cheating

- A CBC survey of Canadian universities shows more than 7,000 students were disciplined for academic cheating in 2011-12, a finding experts say falls well short of the number of students who actually cheat.
- In the first survey of its kind, CBC News contacted 54 universities and asked them to provide the number of 2011-12 academic misconduct cases that went through a formal discipline process.
- Forty-two institutions supplied data, showing less than one per cent of total students were affected.
- "There's a huge gap between what students are telling us they're doing and the numbers of students that are being caught and sanctioned for those behaviours," said Julia Christensen Hughes,
- Hughes said surveys of students show that more than 50 per cent admit to different forms of cheating.

SFU disciplines more cheating students than UBC, survey says

More than 500 students disciplined for academic dishonesty at SFU, only 36 at UBC, between 2011-2012

CBC News Posted: Feb 25, 2014 6:19 AM PT | Last Updated: Feb 25, 2014 6:36 PM PT

Stay Connected with CBC News

Mobile Facebook Podcasts Twitter Alerts Newsletter

SFU

... cheating

- Detecting cheating can be hard. Christensen Hughes published a study in 2006 that found that more than 50 per cent of undergraduate students and 35 per cent of graduate students admitted they had cheated on written work.

CSSHE
SCÉES
Canadian Journal of Higher Education
Revue canadienne d'enseignement supérieur
Volume 36, No. 2, 2006, pages 1 - 21
www.ingentaconnect.com/content/csshe/cjhe

Academic Misconduct within Higher Education in Canada

Julia M. Christensen Hughes
University of Guelph

Donald L. McCabe
Rutgers University

... cheating

- This paper ... presenting the results of a study conducted at 11 Canadian higher education institutions between January 2002 and March 2003.
- A modified version of the survey utilized in the Center for Academic Integrity's Assessment Project ... was used to collect data from 11 Canadian higher education institutions between January 2002 and March 2003
- Each institution was encouraged to advertise the project broadly and an e-mail message inviting participation was distributed to each institution's entire academic population
- Response rates ranged from approximately 5 to 25%
- In addition to these low to modest response rates, this study had several limitations

... cheating

- Substantially fewer graduate students (only 9%) reported having engaged in one or more instances of serious test cheating behaviour,
- while a surprisingly high number (35%) reported having engaged in one or more instances of serious cheating on written work (see Table 3).
- our findings suggest that these rates may be understated as many graduate students (37%) reported they were certain another student had cheated in a test or exam

Semiparametric Regression §10.7

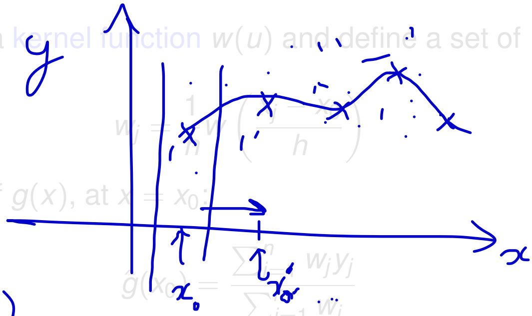
- model $y_j = g(x_j) + \epsilon_j, \quad j = 1, \dots, n \quad x_j$ scalar
- mean function $g(\cdot)$ assumed to be "smooth"

$E(\varepsilon_j) = 0$

y_j ind't

- introduce a kernel function $w(u)$ and define a set of weights

$$w_j = \frac{1}{h} w\left(\frac{x_j - x_0}{h}\right)$$



- estimate of $g(x)$, at $x = x_0$:

$$\hat{g}(x_0) = \frac{\sum_{j=1}^{n} w_j y_j}{\sum_{j=1}^{n} w_j}$$

$E(y \mid x)$

$\bar{y}$(n-hood of x)

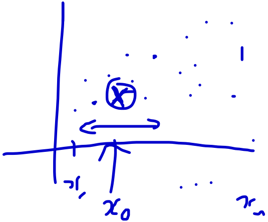
- Nadaraya-Watson estimator (10.40) – local averaging

Semiparametric Regression §10.7

- model $y_j = g(x_j) + \epsilon_j, \quad j = 1, \ldots, n \quad x_j$ scalar
- mean function $g(\cdot)$ assumed to be "smooth"
- introduce a kernel function $w(u)$ and define a set of weights

$w(u) = e^{-\frac{1}{2}u^2}$

$$w_j = \frac{1}{h} w\left(\frac{x_j - x_0}{h}\right)$$



- estimate of $g(x)$, at $x = x_0$:

$$\hat{g}(x_0) = \frac{\sum_{j=1}^{n} w_j y_j}{\sum_{j=1}^{n} w_j}$$

- Nadaraya-Watson estimator (10.40) – local averaging

Semiparametric Regression §10.7

- model $y_j = g(x_j) + \epsilon_j, \quad j = 1, \dots, n \quad x_j$ scalar
- mean function $g(\cdot)$ assumed to be "smooth"
- introduce a kernel function $w(u)$ and define a set of weights
$$w_j = \frac{1}{h} w\left(\frac{x_j - x_0}{h}\right)$$
- estimate of $g(x)$, at $x = x_0$:
$$\hat{g}(x_0) = \frac{\sum_{j=1}^n w_j y_j}{\sum_{j=1}^n w_j}$$
- Nadaraya-Watson estimator (10.40) – local averaging

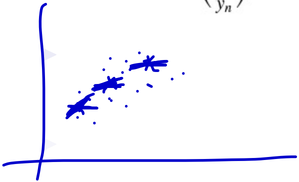
... kernel smoothing

- better estimates can be obtained using local regression at point x

poly of degree k

-

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & (x_1 - x_0) & \cdots & (x_1 - x_0)^k \\ \vdots & \vdots & & \vdots \\ 1 & (x_n - x_0) & \cdots & (x_n - x_0)^k \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix},$$



$$\hat{\beta} = (X^T W X)^{-1} X^T W y$$

$$\hat{g}(x_0) = \hat{\beta}_0$$

- usually obtain estimates $\hat{g}(x_j), j = 1, \ldots, n$

... kernel smoothing

- better estimates can be obtained using local regression at point x

- $$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & (x_1 - x_0) & \cdots & (x_1 - x_0)^k \\ \vdots & \vdots & & \vdots \\ 1 & (x_n - x_0) & \cdots & (x_n - x_0)^k \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix},$$

- $$\hat{\beta} = (X^T W X)^{-1} X^T W y$$

$w_j = w\left(\frac{x_j - x_0}{h}\right) \cdot \frac{1}{h}$

- $$\hat{g}(x_0) = \hat{\beta}_0$$

- usually obtain estimates $\hat{g}(x_j), j = 1, \ldots, n$

... kernel smoothing

- better estimates can be obtained using local regression at point x

$k = 1$

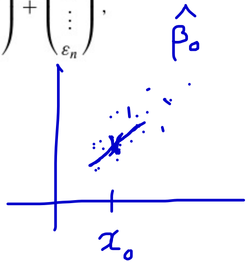
- $$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & (x_1 - x_0) & \cdots & (x_1 - x_0)^k \\ \vdots & \vdots & & \vdots \\ 1 & \underline{(x_n - x_0)} & \cdots & (x_n - x_0)^k \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix},$$

$y = \alpha + \beta x$

$= \beta_0 + \beta(x - x_0)$

- $$\hat{\beta} = (X^T W X)^{-1} X^T W y$$

- $$\hat{g}(x_0) = \hat{\beta}_0$$



- usually obtain estimates $\hat{g}(x_j), j = 1, \ldots, n$

... kernel smoothing

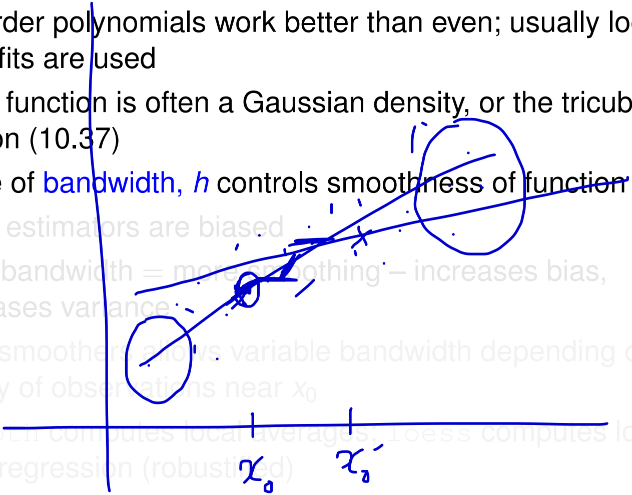
- better estimates can be obtained using local regression at point x
- $$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & (x_1 - x_0) & \cdots & (x_1 - x_0)^k \\ \vdots & \vdots & & \vdots \\ 1 & (x_n - x_0) & \cdots & (x_n - x_0)^k \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix},$$
- $$\hat{\beta} = (X^T W X)^{-1} X^T W y$$
- $$\hat{g}(x_0) = \hat{\beta}_0$$
- usually obtain estimates $\hat{g}(x_j), j = 1, \ldots, n$

... kernel smoothing

- odd-order polynomials work better than even; usually local linear fits are used
- kernel function is often a Gaussian density, or the tricube function (10.37)
- choice of bandwidth, h controls smoothness of function
- kernel estimators are biased
- larger bandwidth $=$ more smoothing – increases bias, decreases variance
- some smoothers allows variable bandwidth depending on density of observations near x_0
- `ksmooth` computes local averages; `loess` computes local linear regression (robustified)

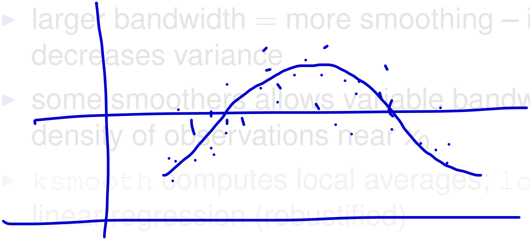
... kernel smoothing

- odd-order polynomials work better than even; usually local linear fits are used
- kernel function is often a Gaussian density, or the tricube function (10.37)
- choice of bandwidth, h controls smoothness of function
- kernel estimators are biased
- larger bandwidth = more smoothing – increases bias, decreases variance
- some smoothers allows variable bandwidth depending on density of observations near x_0
- `ksmooth` computes local averages; `loess` computes local linear regression (robustified)



... kernel smoothing

- odd-order polynomials work better than even; usually local linear fits are used
- kernel function is often a Gaussian density, or the tricube function (10.37)
- choice of bandwidth, h controls smoothness of function
- kernel estimators are biased $E\hat{g}(x_0) \neq g(x_0)$
- larger bandwidth = more smoothing – increases bias, decreases variance
- some smoothers allows variable bandwidth depending on density of observations near x_0
- `ksmooth` computes local averages; `loess` computes local linear regression (robustified)



... kernel smoothing

- odd-order polynomials work better than even; usually local linear fits are used
- kernel function is often a Gaussian density, or the tricube function (10.37)
- choice of bandwidth, h controls smoothness of function
- kernel estimators are biased
- larger bandwidth $=$ more smoothing – increases bias, decreases variance
- some smoothers allows variable bandwidth depending on density of observations near x_0
- `ksmooth` computes local averages; `loess` computes local linear regression (robustified)

... kernel smoothing

- odd-order polynomials work better than even; usually local linear fits are used
- kernel function is often a Gaussian density, or the tricube function (10.37)
- choice of bandwidth, h controls smoothness of function
- kernel estimators are biased
- larger bandwidth $=$ more smoothing – increases bias, decreases variance
- some smoothers allows variable bandwidth depending on density of observations near x_0
- `ksmooth` computes local averages; `loess` computes local linear regression (robustified)

Example: weighted average

```
?ksmooth

ksmooth(x,y,kernel=c("box","normal"),bandwidth=0.5,
         range.x=range(x),
         n.points=max(100,length(x)), x.points)
```

$n=100$

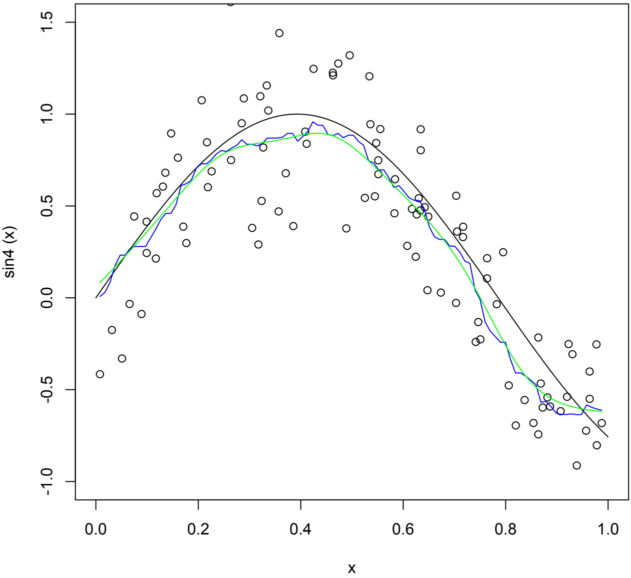
$y_i = \sin(4x_i) + \varepsilon_i$

$x \sim U(0,1)$

$\varepsilon \sim N(0, \frac{1}{3})$

```
> eps<-rnorm(100,0,1/3)
> x<-runif(100)
> sin4 <- function(x){sin(4*x)}
> y<-sin4(x)+eps
> plot(sin4,0,1,type="l",ylim=c(-1.0,1.5),xlim=c(0,1))
> points(x,y)
> lines(ksmooth(x,y,"box",bandwidth=.2),col="blue")
> lines(ksmooth(x,y,"normal",bandwidth=.2),col="green")
```

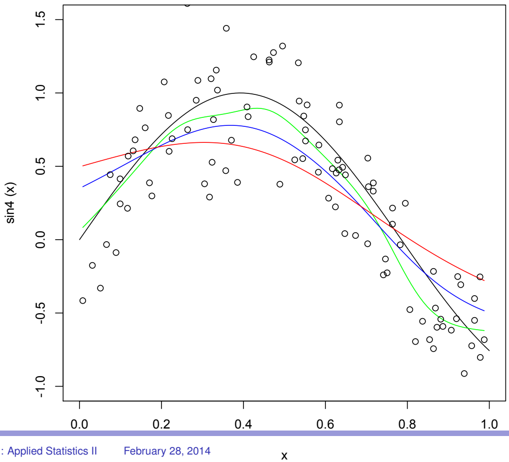
$w(\cdot)$ h



sin4 (x)
x
-1.0
-0.5
0.0
0.5
1.0
1.5
0.0
0.2
0.4
0.6
0.8
1.0

... Example

```
> plot(sin4,0,1,type="l",ylim=c(-1.0,1.5),xlim=c(0,1))
> lines(ksmooth(x,y,"normal",bandwidth=.2),col="green")
> lines(ksmooth(x,y,"normal",bandwidth=0.4),col="blue")
> lines(ksmooth(x,y,"normal",bandwidth=0.6),col="red")
```



Fitting in `R`

- `scatter.smooth` fits a `loess` curve to a scatter plot
- `loess` takes a `family` argument : `family = gaussian` gives weighted least squares using K_λ as weights and `family=symmetric` gives a robust version using Tukey's biweight   same idea, different implem.
- `supsmu` implements "Friedman's super smoother": a running lines smoother with elaborate adaptive choice of bandwidth
- Library `KernSmooth` has `locpoly` for local polynomial fits, and by setting `degree = 0` gives a kernel smooth   kernel regression
- as usual more smoothing means larger bias, smaller variance   also ksmooth   kernel regression

Example: local linear smoothing

```
> plot(sin4,0,1,type="l",ylim=c(-1,1.5),xlim=c(0,1), xlab = "x")
> lo1 = loess(y ~ x, degree = 1, span = 0.75)
```

linear reg

bandw.

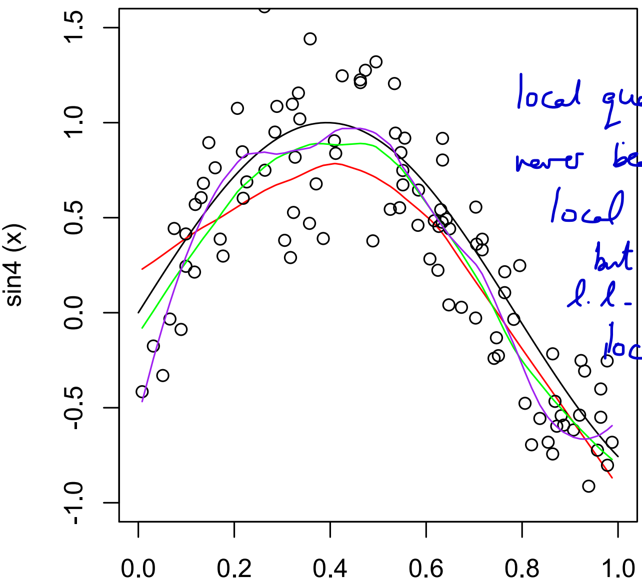
```
> attributes(lo1)
$names
 [1] "n"          "fitted"     "residuals" "enp"        "s"          "one.delta"
 [7] "two.delta" "trace.hat" "divisor"   "pars"       "kd"         "call"
[13] "terms"      "xnames"     "x"          "y"          "weights"

$class
[1] "loess"

> lines(lo1$x[ord],lo1$fitted[ord],col="red")
> lo2 = loess(y~x, degree=1, span=0.4)
> lo3 = loess(y~x, degree=2, span=0.4)
```

smaller span

```
> lines(lo1$x[ord],lo2$fitted[ord],col="green")
> lines(lo1$x[ord],lo3$fitted[ord],col="purple")
```



local quad. regr never beats local linear but l.l. beats local avg.

Scatter Plot with Smooth Curve Fitted by Loess

Description

Plot and add a smooth curve computed by `loess` to a scatter plot.

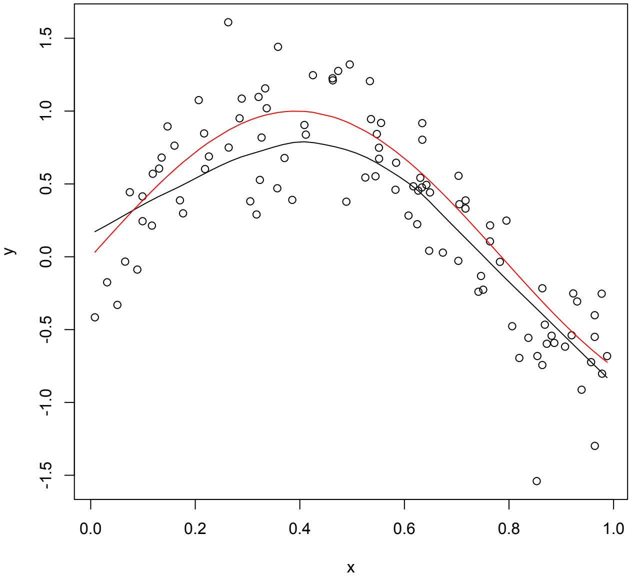
Usage

```
scatter.smooth(x, y = NULL, span = 2/3, degree = 1,
    family = c("symmetric", "gaussian"),
    xlab = NULL, ylab = NULL,
    ylim = range(y, prediction$y, na.rm = TRUE),
    evaluation = 50, ...)

loess.smooth(x, y, span = 2/3, degree = 1,
    family = c("symmetric", "gaussian"), evaluation = 50, ...)
```

Arguments

`x,y`	the `x` and `y` arguments provide the x and y coordinates for the plot. Any reasonable way of the coordinates is acceptable. See the function `xy.coords` for details.
`span`	smoothness parameter for `loess`.
`degree`	degree of local polynomial used.
`family`	if `"gaussian"` fitting is by least-squares, and if `family="symmetric"` a re-descending M is used.
`xlab`	label for x axis.
`ylab`	label for y axis.



1.5
1.0
0.5
0.0
-0.5
-1.0
-1.5
y
0.0
0.2
0.4
0.6
0.8
1.0
x

Friedman's SuperSmoother

Description

Smooth the (x, y) values by Friedman's 'super smoother'.

Usage

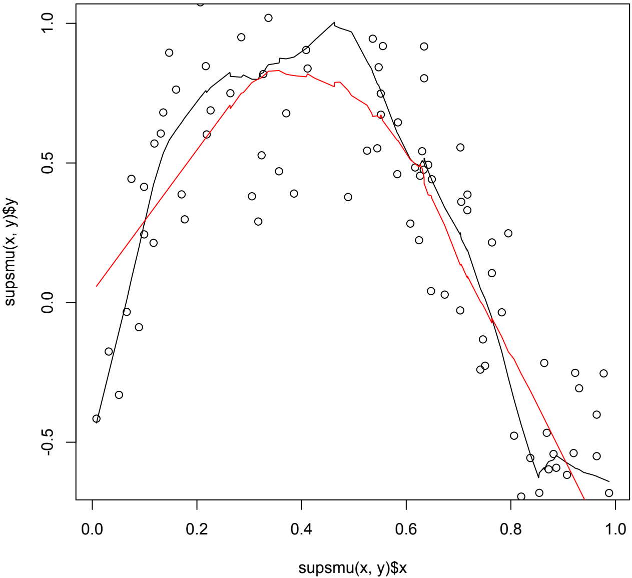
```
supsmu(x, y, wt, span = "cv", periodic = FALSE, bass = 0)
```

Arguments

`x`	x values for smoothing
`y`	y values for smoothing
`wt`	case weights, by default all equal
`span`	the fraction of the observations in the span of the running lines smoother, or `"cv"` to choose th leave-one-out cross-validation.
`periodic`	if `TRUE`, the x values are assumed to be in `[0, 1]` and of period 1.
`bass`	controls the smoothness of the fitted curve. Values of up to 10 indicate increasing smoothness

Details

`supsmu` is a running lines smoother which chooses between three spans for the lines. The running lines sı are symmetric, with `k/2` data points each side of the predicted point, and values of `k` as `0.5 * n`, `0.2 *` `0.05 * n`, where `n` is the number of data points. If `span` is specified, a single smoother with span `span *` used.



1.0
0.5
0.0
-0.5
0.0
0.2
0.4
0.6
0.8
1.0
supsmu(x, y)$y
supsmu(x, y)$x

goodby

FEBRUARY

hello

MARCH