STA547: HW3 [*]

Zhiling Gu

Contents

[*]Instructor: Xiongtao Dai, Iowa State University

1 K-L Representation of Brownian Bridge

Recall K-L theorem

Theorem: Let $\{X(t), t \in \mathcal{T}\}$ be a mean sq cont process. Then

$$\lim_{k\to\infty} \sup_{t\in\mathcal{T}} E(X(t) - X_k(t))^2 = 0$$

where

$$X_K(t) := \mu(t) + \sum_{j=1}^{K} I_{x-\mu}(e_j)e_j(t)$$

alternatively, we say

$$X(t) = \mu(t) + \sum_{j=1}^{\infty} I_{x-\mu}(e_j)e_j(t)$$

where the sum converges in $L^2(\Omega)$ uniformly over $t \in \mathcal{T}$.

$$I_X(f, P) = \sum_{i=1}^{m(n)} X(t_i) \int_{E_i} f(u)du$$

and

$$G(t, s) = \sum_{i=1}^{\infty} \lambda_j e_j(t)t_j(s)$$

Brownian Bridge: $B(t) := (W(t)|W(1) = 0) = W(t) - tW(1), t \in [0, 1]$. First we derive covariance function $G(t, s)$ and mean function $\mu(t)$:

$$\begin{aligned} G(t, s) &= Cov(B(t), B(s)) \\ &= Cov(W(t) - tW(1), W(s) - sW(1)) \\ &= \min(t, s) - 2ts + ts \\ &= \min(t, s) - ts \\ \mu(t) &= E(B(t)) = 0 \end{aligned}$$

Since μ and G are continuous function, given that B is a second order process, we know B is mean square continuous. In addition, we have the eigen function of G is also continuous.

Sencondly find the eigen function and eigen value pair.

$$\begin{aligned} \int_0^1 G(t, s)e(s)ds &= \lambda e(t) \\ \int_0^1 (\min(t, s) - ts)e(s)ds &= \lambda e(t) \\ \int_0^t se(s)ds + \int_t^1 te(s)ds - \int_0^1 tse(s)ds &= \lambda e(t) \end{aligned}$$

Since $e(\cdot)$ is condition, apply the foundamental theorem of calculus, and differentiate both sides with respect to t to the above result, we then have

$$\begin{aligned} te(t) + \int_t^1 e(s)ds - te(t) - \int_0^1 se(s)ds &= \lambda e'(t) \\ \int_t^1 e(s)ds - \int_0^1 se(s)ds &= \lambda e'(t) \end{aligned}$$

Differentiate again on both sides

$$-e(t) = \lambda e''(t)$$

If $\lambda = 0$ then $e(\cdot) = 0$, so 0 cannot be an eigen value. The solution to the above PDE is

$$e(s) = A\sin(s/\sqrt{\lambda}) + B\cos(s/\sqrt{\lambda}),\ \text{for some } A, B \in \mathbb{R}$$

Note $e(0) = 0$, so $B = 0$. In addition, observe

$$\begin{aligned}
\lambda e'(1) &= -\int_0^1 se(s)ds \\
\lambda e''(s) &= -e(s) \\
\implies \lambda e'(1) &= \int_0^1 s\lambda e''(s)ds \\
e'(1) &= \int_0^1 se''(s)ds = e'(1) - \int_0^1 e'(s)ds \\
e(1) &= e(0) \\
\implies A\sin(1/\sqrt{\lambda}) &= 0
\end{aligned}$$

So $1/\sqrt{\lambda} = j\pi$. This shows

$$\lambda_j = 1/(j^2\pi^2) \quad e_j(t) = \sqrt{2}\sin(tj\pi), j = 1, 2, \ldots \tag{1}$$

Since $\mu(t) = 0$, we only need to find $\xi_j(t) := I_B(e_j) = \sum_{i=1}^{m(n)} B(t_i)\int_{E_i} e_j(u)du$. Since $B(t_i)$ is a Gassian random variable, therefore $\xi_j(t)$ is also Gaussian. We can calculate the mean and variance of ξ_j as $E(\xi_j(t)) = 0, Var(\xi_j(t)) = E(\xi_j^2(t)) = \lambda_j$, therefore $\xi_j \sim N(0, 1/(j^2\pi^2))$. Then we can finally write $B(t)$ in the KL represention as

$$B(t) = 0 + \sqrt{2}\sum_{j=1}^{\infty}\xi_j\sin(\pi jt) = 0 + \sqrt{2}\sum_{j=1}^{\infty} Z_j\sin(\pi jt)/(j\pi)$$

✓

where $Z_j \sim N(0, 1)$ and the convergence holds for each $t \in [0, 1]$ in $L^2(\Omega)$. $\square$

2 Simulation of Brownian Motion and Brownian Bridge

Similarly we are able to get K-L expansion of brownian motion $X(t), t \in [0, 1]$ as

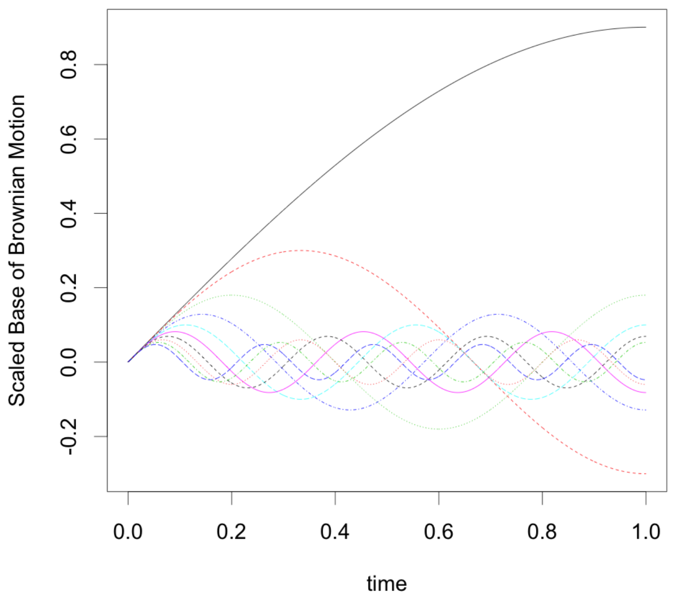
$$X(t) = \sqrt{2}\sum_{j=1}^{\infty} Z_j\sin((j-1/2)\pi t)/((j-1/2)\pi)$$

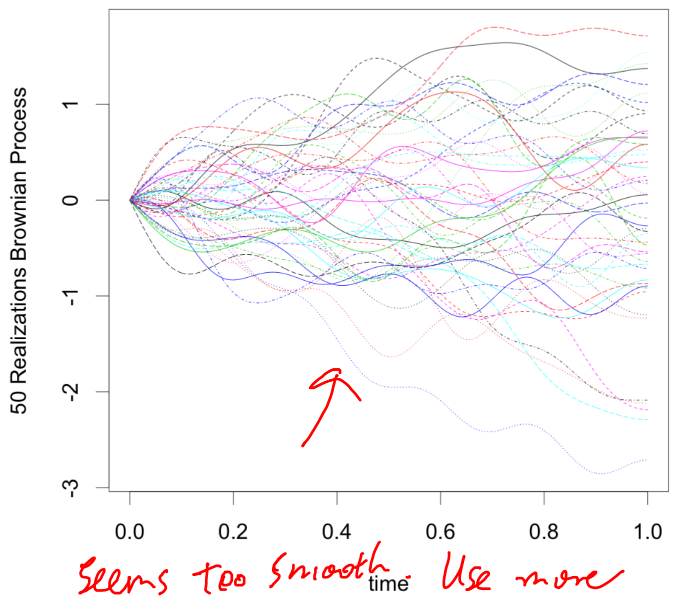
where $Z_j \sim N(0, 1)$ and the convergence holds for each $t \in [0, 1]$ in $L^2(\Omega)$.

See the program and the simulated trajectories as following

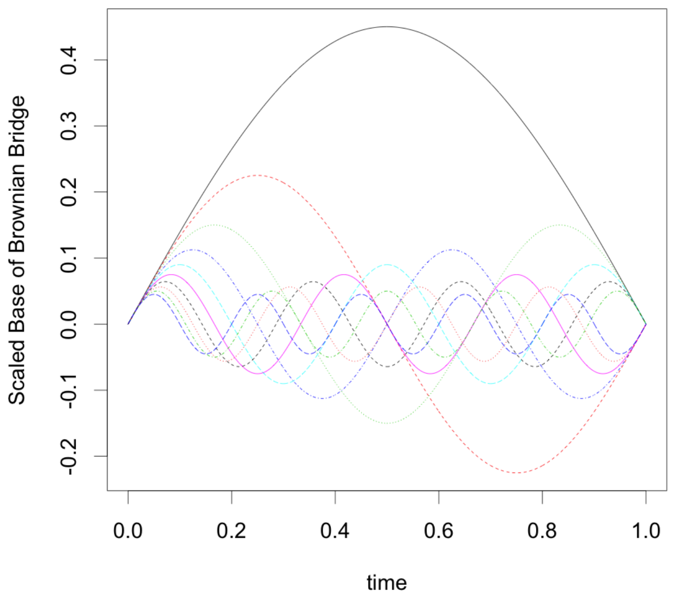
```r
n.time <- 1000 # number of points on each trajectory
n.real <- 50 # number of realized trajectories
n.base <- 10 # number of basis
T <- seq(0,1, length.out = n.time)
Z <- matrix(rnorm(n.time * n.real), ncol = n.real, nrow = n.base)


base.X<- function(t){
    1/ (((1:n.base)-.5)*pi) * sqrt(2) * sin(pi * t * ((1:n.base)-.5))
}

base.B<- function(t){
    1/ ((1:n.base)*pi) * sqrt(2) * sin(pi * t * (1:n.base))
}

# ---------------------------
# Brownian motion simulation
# ---------------------------
X.base <- t(sapply(T, base.X))   # every column is a scaled base on [0,1]
matplot(T, X.base, type='l', xlab = "time", ylab ="Scaled Base of Brownian Motion")
X <-  X.base %*% Z
```

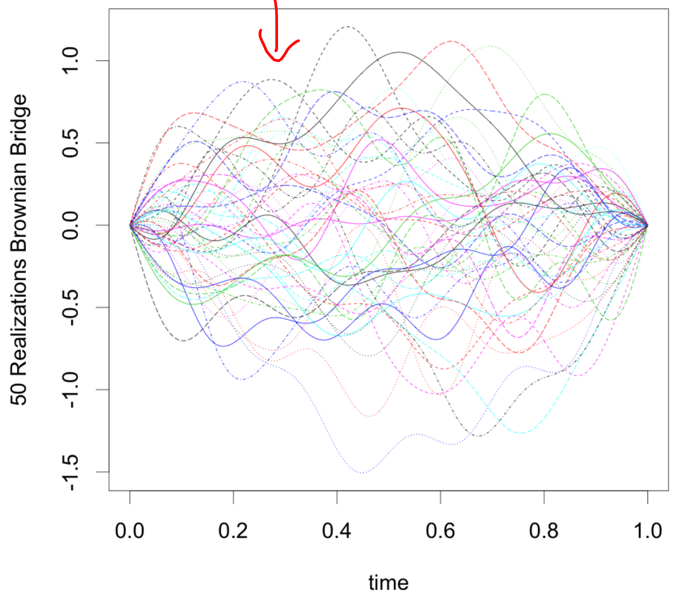
```r
matplot(T, X, type='l', xlab = "time", ylab ="50 Realizations Brownian Process")
dev.off()

# ----------------------------
# Brownian bridge simulation
# ----------------------------

B.base <- t(sapply(T, base.B))
matplot(T, B.base, type='l', xlab = "time", ylab ="Scaled Base of Brownian Bridge")
B <-  B.base %*% Z
matplot(T, B, type='l', xlab = "time", ylab ="50 Realizations Brownian Bridge")
```





Seems too smooth. Use more components





3 Kernel Density Estimation

Let $X_1, \ldots, X_n$ be a sample of i.i.d. real-valued random variables sharing a distribution with an unknown density f supported on a compact interval $[a, b]$. The kernel density estimate (KDE) of $f(x_0)$ at $x_0 \in [a, b]$ is

$$\hat{f}(x_0) = \frac{1}{n} \sum_{i=1}^{n} K_h(X_i - x_0)$$

where $K_h(\cdot) = h^{-1}K(\cdot/h), K(\cdot)$ is a kernel function, $h > 0$ is the bandwidth. Write down an intuitive argument for why the KDE "works'. [Hint: consider a uniform kernel K]

Proof. Consider $K(x) = \frac{1}{2}1_{[-1,1]}(x)$, then $K_h(x) = \frac{1}{2h}1_{[-1,1]}(x/h)$. Further

$$\begin{aligned}
E_X(\hat{f}(x_0)) &= E_X(\frac{1}{n}\sum_{i=1}^{n} K_h(X_i - x_0)) \\
&= E_X(K_h(X_i - x_0)) \\
&= E_X\left(\frac{1}{2h}1_{[-1,1]}((X_i - x_0)/h)\right) \\
&= \frac{1}{2h}\int_{\mathcal{X}} 1_{(x_0-h,x_0+h)}(x)f(x)dx \\
&= \frac{1}{2h}\int_{x_0-h}^{x_0+h} f(x)dx \\
&\to f(x_0) \text{ as } h \to 0
\end{aligned}$$

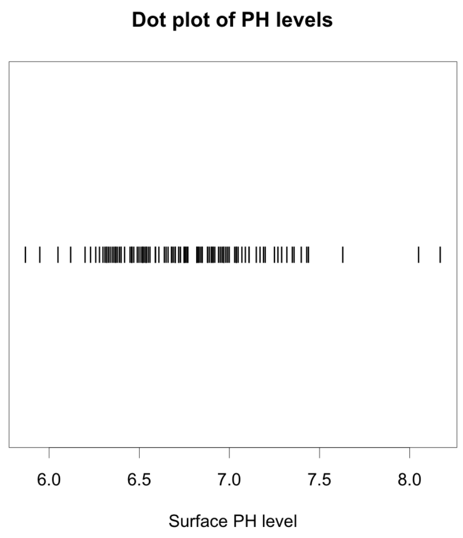
The integral divided by the bandwidth approaches the density when bandwidth goes to zero. □

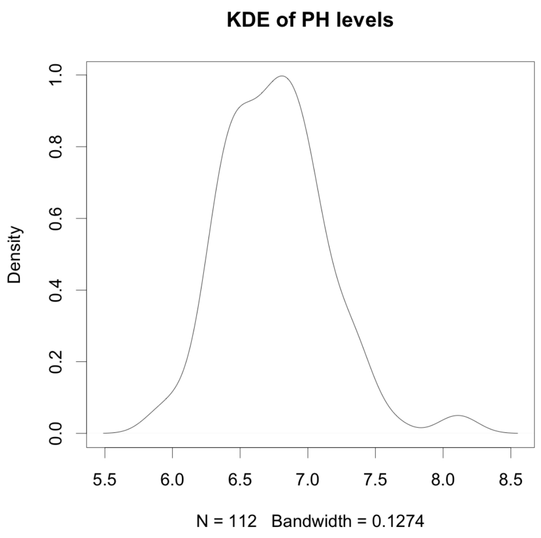
4 Nonparametric regression

Analyze the Lake Acidity data in the gss package of R. The data were extracted from the Eastern Lake Survey of 1984 conducted by the United States Environmental Protection Agency, concerning 112 lakes in the Blue Ridge. To gain access to the data, type the following commends in R: library (gss) data (LakeAcidity)

The data set has 5 variables for each there are 112 observationsPerform a nonparametric regression on the calcium concentration (Y) against surface ph level (X).

(a) Show a KDE and a dot plot of the ph levels.





(b) Compare the results of local polynomial estimator, smoothing spline, regression spline and penalized spline. Manually vary the tuning parameters, including bandwidth, the number of knots, and the penalty λ on the second derivative of the regression curve. For each smoother identify a parameter setting that

i) over-smooths (the estimate is too smooth),

ii) undersmooths (the estimate is too rough), and

iii) smooths appropriately. Show the graphs and your code.

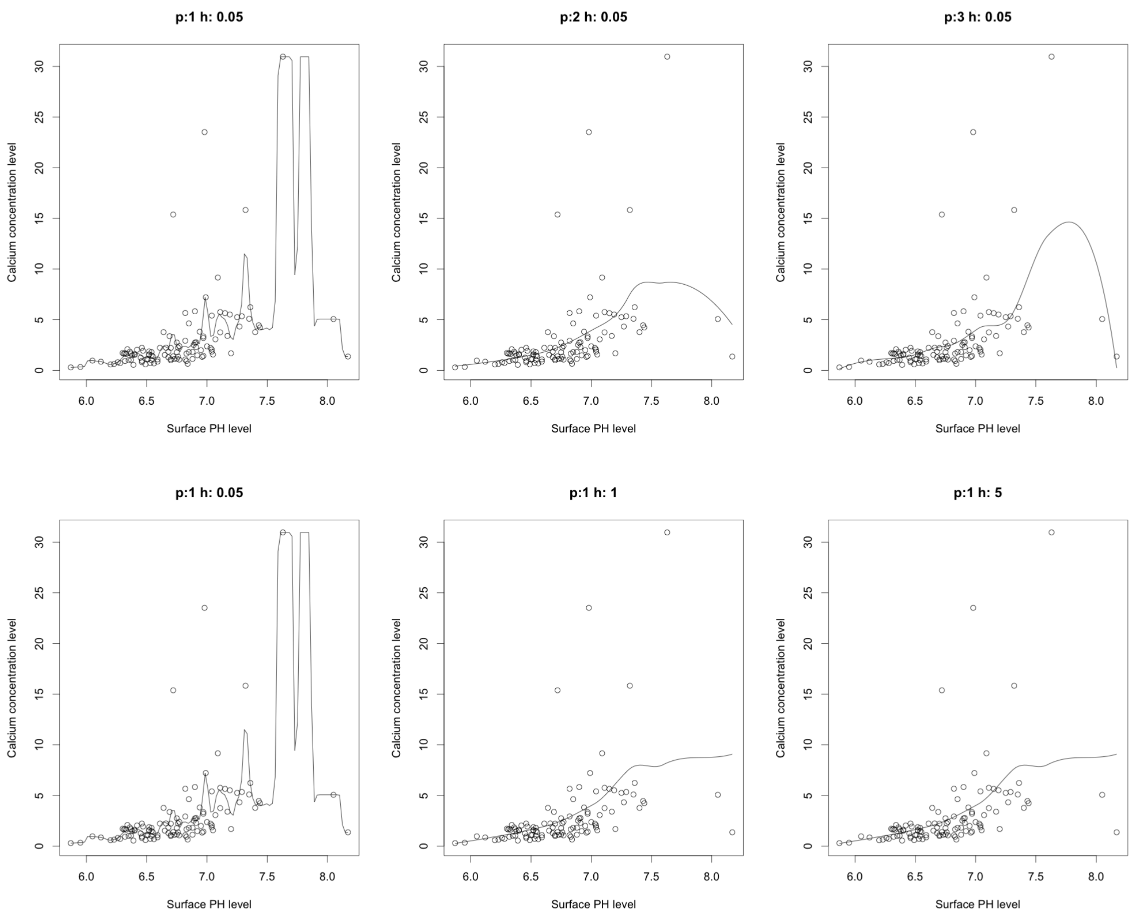
1. Local polynomial:

$$\mu(t) \approx \beta_0 + \sum_{j=1}^{p} (X - X_0)^j \beta_j$$

in each local window $X_0 \pm h$.

High bandwidth h leads to oversmoothing.

- Oversmooth: $p = 1, h = 5$
- Undersmooth: $p = 1, h = .05$
- Appropriate: $\underline{p} = 3, h = .05$ usually one times the BW

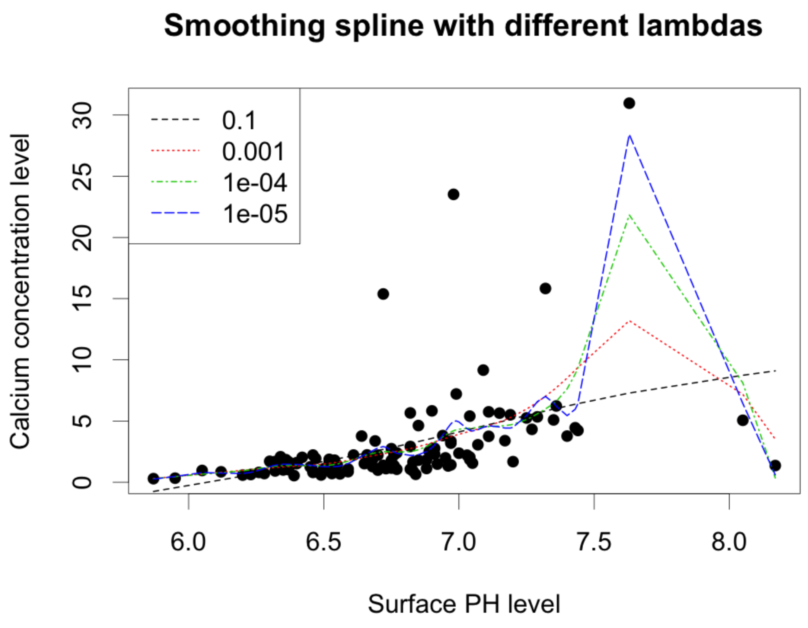


2. Smoothing spline:

$$\arg \min_{f \in S(d,x)} \sum_{i=1}^{n} (Y_i - f(X_i))^2 + \lambda \int_0^1 [f^{(m)}(x)]^2 dx$$

Higher penalty parameter λ leads to oversmoothing.

- Oversmooth: $\lambda = .1$
- Undersmooth: $\lambda = 10^{-5}$
- Appropriate:$\lambda = 10^{-4}$



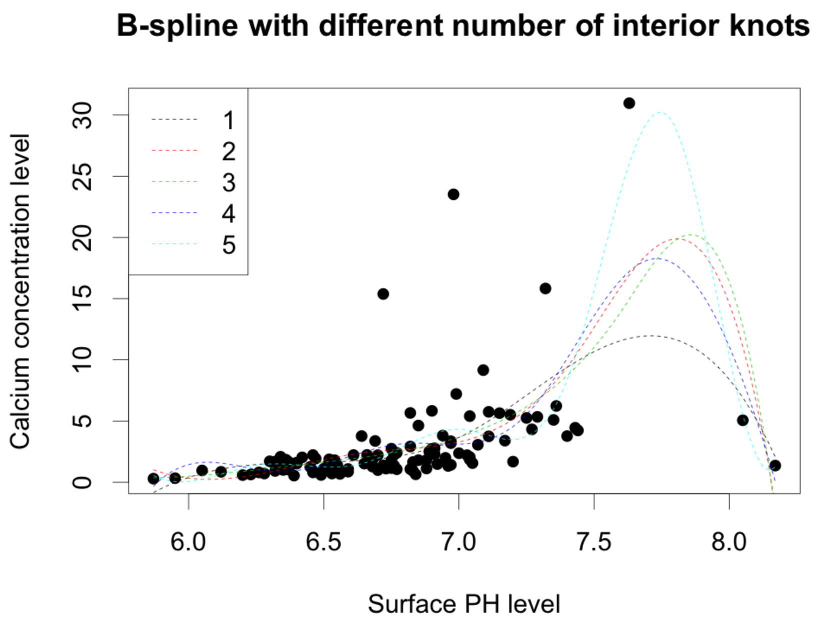
3. Regression spline: defined on $[0, 1]$ with interior knots $0 < \mathcal{K}_1 < \cdots < \mathcal{K}_J < 1$: The $\mathcal{K}_j$ are commonly equally spaced. On knot set $\mathcal{K} = [\mathcal{K}_1 \cdots \mathcal{K}_J]$, a spline function $S(x)$ of order d is a polynomial of degree $(d-1)$ between two adjacent knots, and is $(d-2)$ times continuously differentiable on the knots. Here $d = 2$ means the splines are merely continous at the knots.

   The collection of all order d splines on knots $\mathcal{K}$ is a (functional) vector space denotes $S(d, \mathcal{K})$. To calculate the dimension: on each $[\mathcal{K}_i, \mathcal{K}_{i+1}]$, the polynomial function has d degrees of freesom, and the 0-th, ..., (d-2)-th derivative at each of the J knots must match up.

$$\dim(S(d, \mathcal{K})) = d \times (J + 1) - (d - 1) \times J = d + J \tag{2}$$

   In this question B-spline is applied, we show that low J leads to oversmoothing.

   - Oversmooth: $J = 1$
   - Undersmooth: $J = 5$
   - Appropriate: $J = 2, 3, 4$

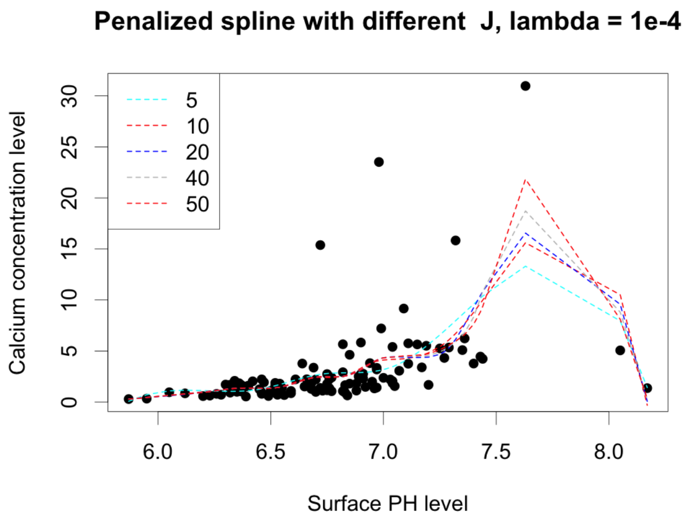


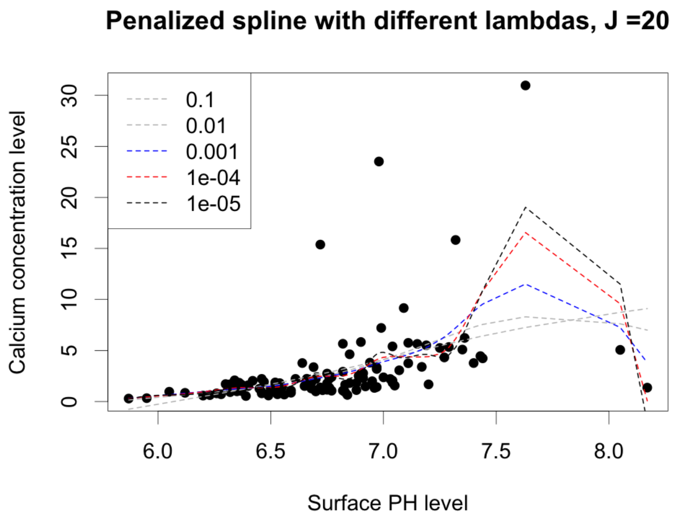
4. Penalized spline:

$$\arg\min_{f \in S(d,\mathcal{K})} \sum_{i=1}^{n} (Y_i - f(X_i))^2 + \lambda \int_0^1 [f^{(m)}(x)]^2 dx$$

See from the graph that low J and high λ leads to oversmoothing.

- Oversmooth: $J = 20, \lambda = .1$
- Undersmooth: $J = 5, \lambda = 10^{-4}$
- Appropriate: $J = 20, \lambda = 10^{-4}$





(c) Write a brief summary of your data analysis.

Comparing the four nonparametric regression methods, we know that the smoothing, regression and penalized spline in general performs better than the local polynomial in the sense that the fitted function are more robust to outliers. Among the four, the penalized spline is the most stable one with respect to the tuning parameters. On the other hand, the calculation for penalized spline is the most demanding. In addition, we also see the local polynomial spline and regression spline are smoother compared to smoothing spline and penalized spline due to their design.

Smoothing splines is usually slower

Note for smoothing spline, regression spline and penalized spline, degree are all taken to be 3. Due to the page limit, please refer to the "s547-hw3-zlgu.R" for the program for question 4.

A