

Math 504 HW12

Jeff Gould

4/9/2020

2

a

$$\min_{\alpha} \sum_{i=1}^N |y_i - \sum_{j=1}^{K-4} \alpha_j b_j(x_i)|^2 + \rho \int_{x_{\min}}^{x_{\max}} \left(\sum_{j=1}^{K-4} \alpha_j b_j(z) \right)''^2 dz \quad (1)$$

From $\sum_{j=1}^{K-4} \alpha_j b_j(x_i)$, we have $N \times K - 4$ values from $b_j(x_i)$. Let each $b_j(x_i)$ be the ij^{th} entry in the matrix B

Also, we can expand $\sum_{j=1}^{K-4} \alpha_j b_j(x_i)$ as $\alpha_1 b_1(x_i) + \alpha_2 b_2(x_i) + \dots + \alpha_{K-4} b_{K-4}(x_i) = S(x_i)$. Thus, we can re-write

$$\sum_{i=1}^N |y_i - \sum_{j=1}^{K-4} \alpha_j b_j(x_i)|^2 = \sum_{i=1}^N |y_i - S(x_i)|^2 = \|y - B\alpha\|^2$$

where B is the $N \times K - 4$ matrix as described above and α is $K - 4$ column vector.

Next, look at $\rho \int_{x_{\min}}^{x_{\max}} \left(\sum_{j=1}^{K-4} \alpha_j b_j(z) \right)''^2 dz$. Since ρ is a constant, we can ignore it for now. This leaves

$$\begin{aligned} \int_{x_{\min}}^{x_{\max}} \left(\sum_{j=1}^{K-4} \alpha_j b_j''(z) \right)^2 dz &= \int_{x_{\min}}^{x_{\max}} \left(\left(\sum_{j=1}^{K-4} \alpha_j b_j''(z) \right) \left(\sum_{i=1}^{K-4} \alpha_i b_i''(z) \right) \right) dz = \\ &= \int_{x_{\min}}^{x_{\max}} \sum_{j=1}^{K-4} \sum_{i=1}^{K-4} \alpha_j \alpha_i b_j''(z) b_i''(z) dz = \sum_{j=1}^{K-4} \sum_{i=1}^{K-4} \alpha_j \alpha_i \left(\int_{x_{\min}}^{x_{\max}} b_j''(z) b_i''(z) dz \right) \end{aligned}$$

Now define a $K - 4$ square matrix Ω such that each ji^{th} corresponds to $\int_{x_{\min}}^{x_{\max}} b_j''(z) b_i''(z) dz$. Then:

$$\sum_{j=1}^{K-4} \sum_{i=1}^{K-4} \alpha_j \alpha_i \left(\int_{x_{\min}}^{x_{\max}} b_j''(z) b_i''(z) dz \right) = \sum_{j=1}^{K-4} \sum_{i=1}^{K-4} \alpha_j \alpha_i \Omega_{ji} = \alpha^T \Omega \alpha$$

Putting this together, we get

$$\min_{\alpha} \sum_{i=1}^N |y_i - \sum_{j=1}^{K-4} \alpha_j b_j(x_i)|^2 + \rho \int_{x_{\min}}^{x_{\max}} \left(\sum_{j=1}^{K-4} \alpha_j b_j(z) \right)''^2 dz = \min_{\alpha} \|y - B\alpha\|^2 + \rho \alpha^T \Omega \alpha \quad (2)$$

Solve for α :

$$L(\alpha) = \|y - B\alpha\|^2 + \rho \alpha^T \Omega \alpha = (y - B\alpha) \cdot (y - B\alpha) + \rho \alpha^T \Omega \alpha = y \cdot y - 2B\alpha \cdot y + B\alpha \cdot B\alpha + \rho \alpha^T \Omega \alpha \Rightarrow$$

$$\begin{aligned}\nabla_{\alpha} L(\alpha) &= -2B^T y + 2B^T B\alpha + 2\rho\Omega\alpha = 0 \longrightarrow B^T B\alpha + \rho\Omega\alpha = B^T y \rightarrow \\ (B^T B + \rho\Omega)^{-1}(B^T B + \rho\Omega)\alpha &= (B^T B + \rho\Omega)^{-1}B^T y \Rightarrow \\ \alpha &= (B^T B + \rho\Omega)^{-1}B^T y\end{aligned}$$

b

```
bone_mass <- read_delim("BoneMassData.txt", delim = " ")
females <- bone_mass %>% filter(gender == "female")
K <- 1000
min_x <- min(females$age)
max_x <- max(females$age)
myknots = seq(min_x, max_x, length.out = 1002)[2:1001]

h <- (max_x - min_x) / 2500
mygrid <- seq(min_x, max_x, h)

B <- splineDesign(knots = myknots, x = females$age, outer.ok = T)
Bpp <- splineDesign(knots = myknots, x = mygrid, derivs = 2, outer.ok = T)

Omega <- matrix(0, nrow = K-4, ncol = K-4)

test <- sum(Bpp[,10] * Bpp[,15] * h)

for(i in 1:(K-4)){
  for(j in 1:(K-4)){
    Omega[i,j] = sum(Bpp[,i] * Bpp[, j] * h)
  }
}
```

c

Show that $B^T B$ is not invertible - simply show $\det B^T B = 0$:

```
det(t(B) %*% B)
```

```
## [1] 0
```

But $B^T B + \rho\Omega$ is invertible for $\rho > 0$ (show $\det \neq 0$):

```
rho = 3e-06
det(t(B)%*%B + rho * Omega)
```

```
## [1] 6.719545e+19
```

d

```

solveAlpha <- function(rho, b = B, omega = Omega, y = females$spnbmd){
  alpha <- solve(t(b) %*% b + rho * omega) %*% t(b) %*% y
  return(alpha)
}
alpha.01 <- solveAlpha(rho = 0.01)
alpha1 <- solveAlpha(rho = 1)
alpha100 <- solveAlpha(rho = 100)

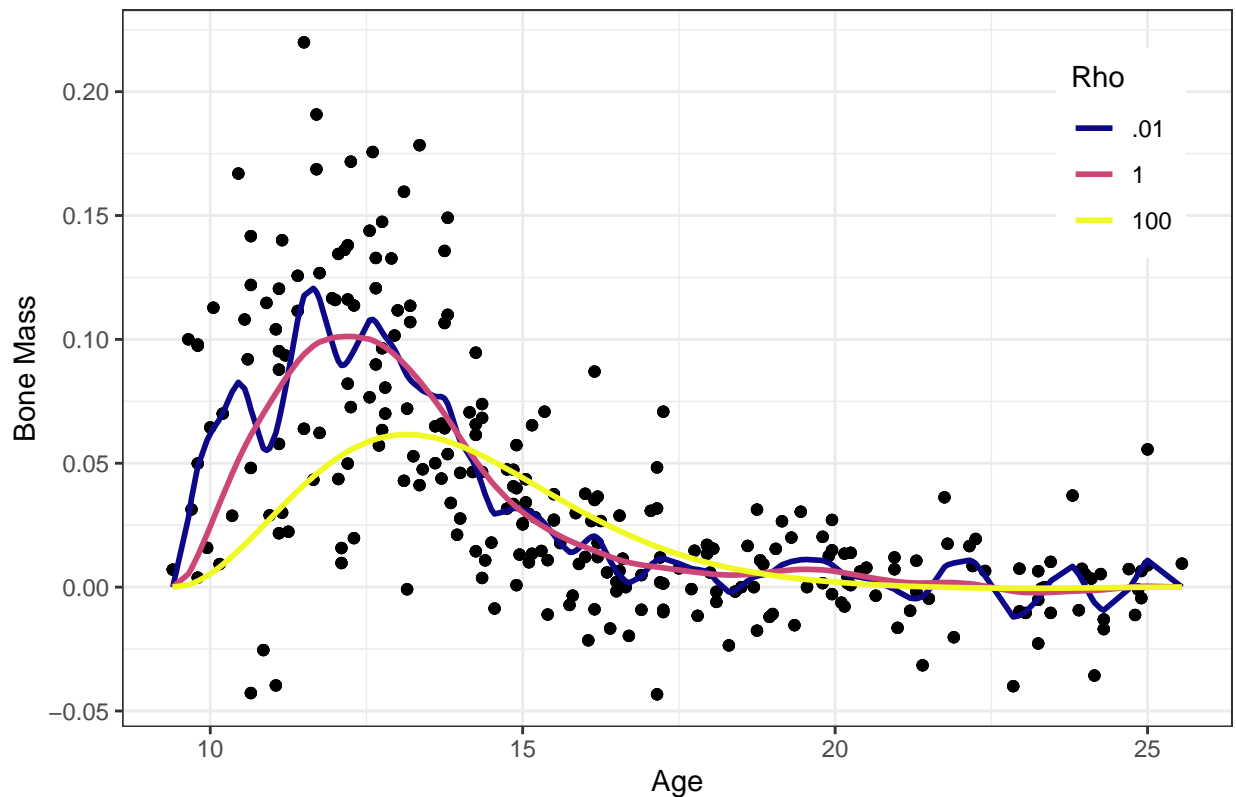
spline_plot_data <- data.frame(x = females$age, boneMass = females$spnbmd,
                              `01` = B %*% alpha.01,
                              `1` = B %*% alpha1,
                              `100` = B %*% alpha100)

spline_test <- spline_plot_data %>%
  pivot_longer(cols = 3:5,
               names_to = "Rho",
               values_to = "S_x") %>%
  mutate(Rho = str_extract(Rho, "\\.[[:digit:]]+|[:digit:]]+"))

ggplot(data = spline_test, aes(x = x)) +
  geom_point(aes(y = boneMass)) +
  geom_line(aes(y = S_x, color = Rho), size = 1) +
  scale_color_viridis(discrete = T, option = "C") +
  theme_bw() +
  labs(title = "1000 Knot Spline Regression with Varying Penalties",
       x = "Age",
       y = "Bone Mass") +
  theme(legend.position = c(0.9,0.8))

```

1000 Knot Spline Regression with Varying Penalties



So we see that using too small of a ρ (blue-ish purple line) causes the regression to be overfit, with the regression line oscillating heavily in small intervals. A spline with a modest penalty of $\rho = 1$ (magenta line), causes the spline regression to look similar to the one computed with two knots on our previous HW. It has a higher rise with the spike in data around ages 11-13 than a higher penalty, but it doesn't have heavy swings the way a lower penalty did. Meanwhile a highly penalized regression with $\rho = 100$ (yellow line), the regression had smaller moves with the data, and perhaps penalized too much. It seems to have too small of a rise with the data from 11-13, and then too slow to drop with the data after age 15.

3

a

signal.R:

```
set.seed(1236)

# the "signal" that make up the rows of our matrix, with noise added.
sig <- seq(1,2,length.out = 10)
sig[5] <- -1

# each row of A will have the form: q*signal + noise
nsamples <- 500
A <- matrix(NA, ncol=10, nrow=nsamples)
```

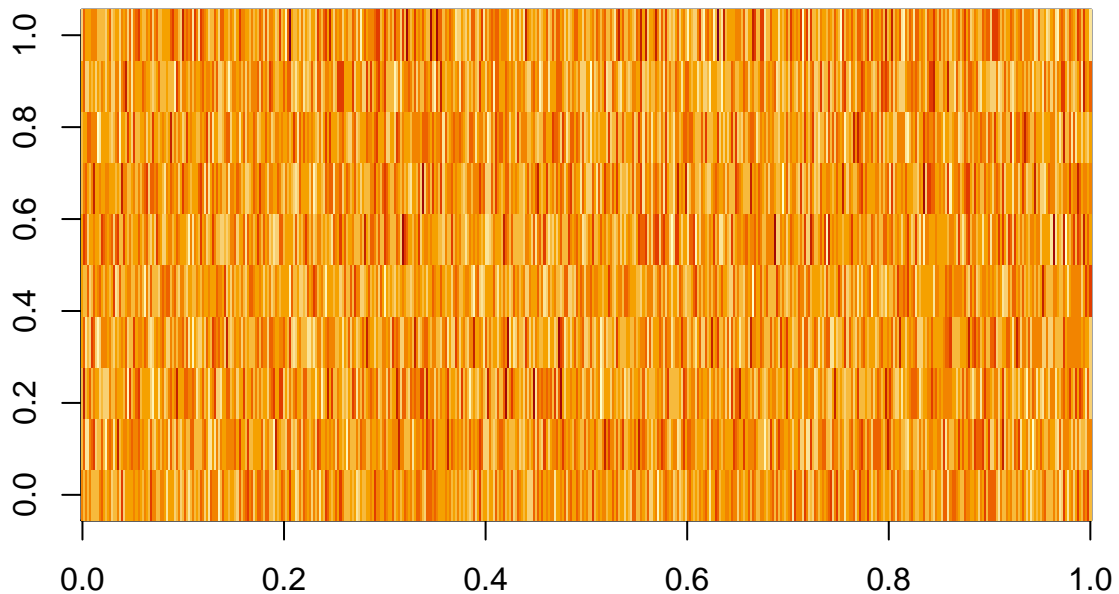
```

no_noise_A <- matrix(NA, ncol=10, nrow=nsamples)
q <- runif(nsamples, min=-3, max=3)
for (i in 1:nsamples) {
  noise <- rnorm(10, mean=0, sd=5)
  A[i,] <- q[i]*sig + noise
  no_noise_A[i,] <- q[i]*sig
}

write.table(no_noise_A, file="no_noise_A.txt", row.names = F, col.names = F)
write.table(A, file="A.txt", row.names = F, col.names = F)
write.table(q, file="q.txt", row.names=F, col.names=F)

image(A)

```



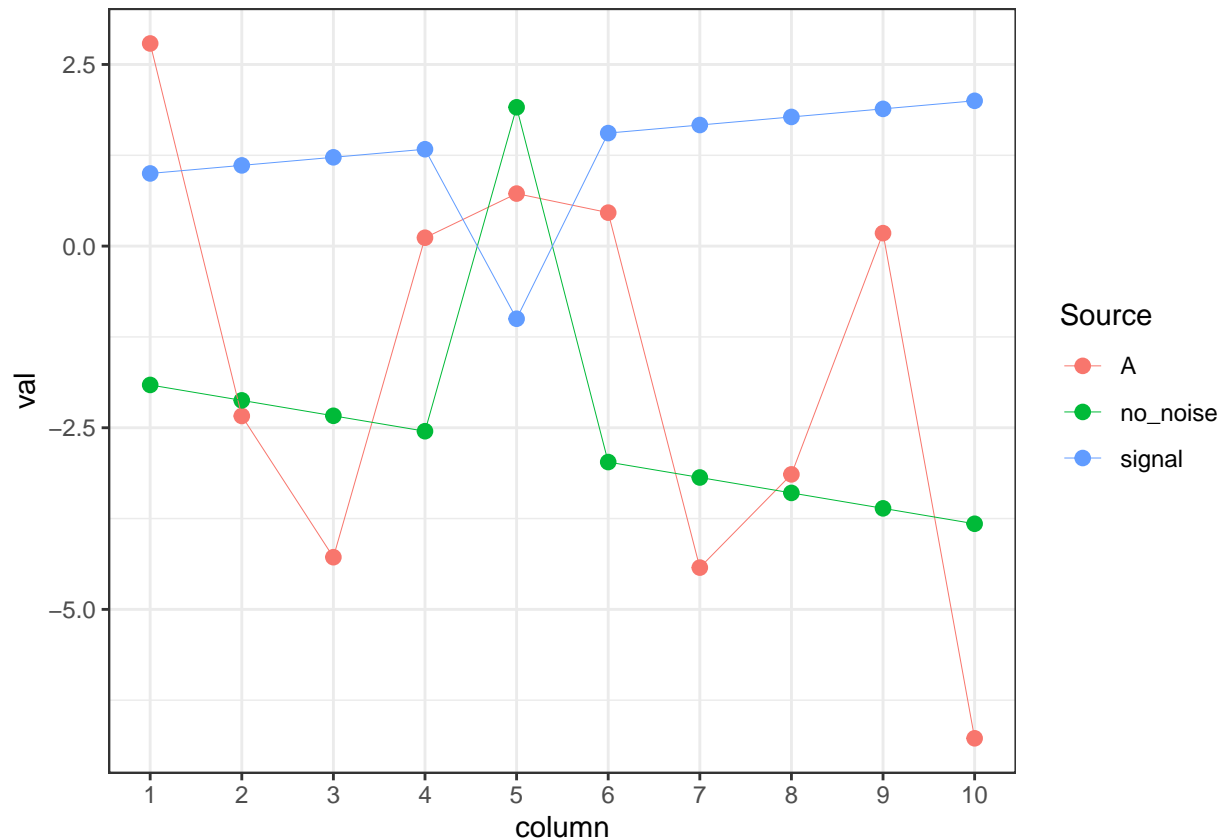
```

plot_data <- data.frame(
  column = seq(1,10,1),
  A = A[1,],
  no_noise = no_noise_A[1,],
  signal = sig
) %>%
  pivot_longer(cols = 2:4, names_to = "Source", values_to = "val")

ggplot(data = plot_data, aes(x = column, y = val)) +
  geom_point(aes(col = Source), size = 2.25) +

```

```
geom_line(aes(col = Source), size = .125) +
scale_x_continuous(breaks = seq(1,10,1), minor_breaks = NULL) +
theme_bw()
```



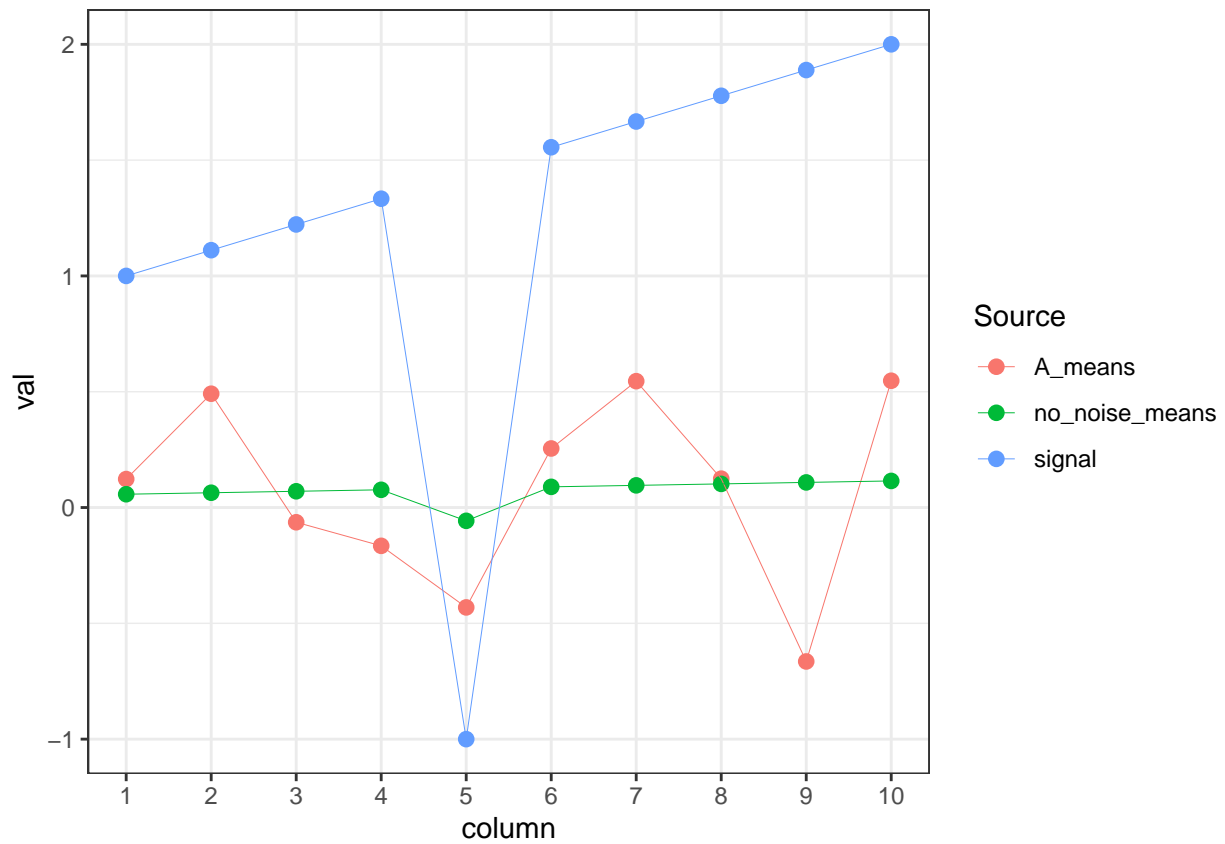
You can't derive the signal just using one observation of *A* and *no_noise_A*, as at a single row observation level the noise is too strong and causes *A* to not follow a tight enough window to derive the noise. As the first row of *no_noise_A* is derived from multiplying $q_1 = -1.911$ times the signal, we would expect to be able to derive the signal by dividing the first row by -1.911 , but doing so with *A* does not give a result reminiscent of our signal: -1.4594568, 1.2235123, 2.2406375, -0.0602157, -0.3782986, -0.2410538, 2.3155057, 1.6442113, -0.093381, 3.5449524

Can we find the signal in *A* using the column means though?

```
col_means_A <- colMeans(A)
col_no_noise_A <- colMeans(no_noise_A)
plot_data_2 <- data.frame(
  column = seq(1,10,1),
  A_means = col_means_A,
  no_noise_means = col_no_noise_A,
  signal = sig
) %>%
  pivot_longer(cols = 2:4, names_to = "Source", values_to = "val")

ggplot(data = plot_data_2, aes(x = column, y = val)) +
  geom_point(aes(col = Source), size = 2.25) +
  geom_line(aes(col = Source), size = .125) +
```

```
scale_x_continuous(breaks = seq(1,10,1), minor_breaks = NULL) +  
theme_bw()
```



By taking the column means we can see the muted signal in **no_noise_A**, however the signal in *A* is still suppressed by the random noise added to the matrix generation.