mlspline for fnc

Choi TaeYoung

2019 12 20

```
## setting
ag2 <- mlsplines::generate_response_smooth(J=50, mod="glm", e_sigma=2, N_s=50)
ag22 <- mlsplines::generate_response_smooth(J=50, mod="glm", e_sigma=2, N_s=100)
ag4 <- mlsplines::generate_response_smooth(J=50, mod="glm", e_sigma=4, N_s=50)
ag44 <- mlsplines::generate_response_smooth(J=50, mod="glm", e_sigma=4, N_s=100)
ag8 <- mlsplines::generate_response_smooth(J=50, mod="glm", e_sigma=8, N_s=50)
ag88 <- mlsplines::generate_response_smooth(J=50, mod="glm", e_sigma=8, N_s=100)

al2 <- mlsplines::generate_response_smooth(J=50, mod="lm", e_sigma=2, N_s=50)
al22 <- mlsplines::generate_response_smooth(J=50, mod="lm", e_sigma=2, N_s=100)
al4 <- mlsplines::generate_response_smooth(J=50, mod="lm", e_sigma=4, N_s=50)
al44 <- mlsplines::generate_response_smooth(J=50, mod="lm", e_sigma=4, N_s=100)
al8 <- mlsplines::generate_response_smooth(J=50, mod="lm", e_sigma=8, N_s=50)
al88 <- mlsplines::generate_response_smooth(J=50, mod="lm", e_sigma=8, N_s=100)


bg2 <- mlsplines::generate_response(J=50, mod="glm", e_sigma=2, N_s=50)
bg22 <- mlsplines::generate_response(J=50, mod="glm", e_sigma=2, N_s=100)
bg4 <- mlsplines::generate_response(J=50, mod="glm", e_sigma=4, N_s=50)
bg44 <- mlsplines::generate_response(J=50, mod="glm", e_sigma=4, N_s=100)
bg8 <- mlsplines::generate_response(J=50, mod="glm", e_sigma=8, N_s=50)
bg88 <- mlsplines::generate_response(J=50, mod="glm", e_sigma=8, N_s=100)

bl2 <- mlsplines::generate_response(J=50, mod="lm", e_sigma=2, N_s=50)
bl22 <- mlsplines::generate_response(J=50, mod="lm", e_sigma=2, N_s=100)
bl4 <- mlsplines::generate_response(J=50, mod="lm", e_sigma=4, N_s=50)
bl44 <- mlsplines::generate_response(J=50, mod="lm", e_sigma=4, N_s=100)
bl8 <- mlsplines::generate_response(J=50, mod="lm", e_sigma=8, N_s=50)
bl88 <- mlsplines::generate_response(J=50, mod="lm", e_sigma=8, N_s=100)
```

```
#lambda
lam1 <- seq(100, 1000, by=1)
lam2 <- seq(10, 100, by=0.1)
lam3 <- seq(1, 10, by=0.01)
lam4 <- seq(0.1, 1, by=0.001)
lam5 <- seq(0.01, 0.1, by=0.0001)
lam6 <- seq(0.001, 0.01, by=0.00001)
lam7 <- seq(0.0001, 0.001, by=0.000001)
lam8 <- seq(0.00001, 0.0001, by=0.0000001)
lam9 <- seq(0.000001, 0.00001, by=0.00000001)
lam0 <- seq(0.0000001, 0.000001, by=0.000000001)

lambda <- c(1000, 100, 10, 1, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001)

lambda <- c(100000, 10000, 1000, 100, 10, 1, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001)
```

```
# Single RMSE (mean of itr=200) ag2
```

```
lambda <- c(1000, 100, 10, 1, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001)
lam <- NULL
RMSE_Single <- NULL

for(j in 1:200){
  GCV_vec <- NULL
  al8 <- mlsplines::generate_response_smooth(J=50, mod="lm", e_sigma=8, N_s=50)

  beta_hat <- NULL
  for(m in 1:50){
    results <- mlsplines::granular(unlist(al8$x_list[m]), unlist(al8$y[m]), mod = "lm")
    beta_hat <- rbind(beta_hat,results)
  }

  K <- mlsplines::make_K(al8$z)

  beta_hat_vec <- unlist(beta_hat[,1])
  RMSE_Single[j] <- rbind(sqrt((1/50)*t(al8$true_mu- beta_hat_vec)%*%(al8$true_mu- beta_hat_vec)))
  }

smRMSE <- mean(RMSE_Single)
ssdRMSE <- sd(RMSE_Single)

# multilevel RMSE (mean of itr=200) al8

lam <- NULL
RMSE_Multilevel <- NULL

for(j in 1:200){
  GCV_vec <- NULL
  al8 <- mlsplines::generate_response_smooth(J=50, mod="lm", e_sigma=8, N_s=50)

  beta_hat <- NULL
  for(m in 1:50){
    results <- mlsplines::granular(unlist(al8$x_list[m]), unlist(al8$y[m]), mod = "lm")
    beta_hat <- rbind(beta_hat,results)
  }

  K <- mlsplines::make_K(al8$z)

    for(i in 1:length(lambda)){
      EM_out <- mlsplines::main_EM(beta_hat_vec = unlist(beta_hat[,1]), V = diag(unlist(beta_hat[,2])),
      GCV_vec <- rbind(GCV_vec,EM_out$GCV)
    }

  lam[j] <- rbind(lambda[which.min(GCV_vec)])
  EM_out <- mlsplines::main_EM(beta_hat_vec = unlist(beta_hat[,1]), V = diag(unlist(beta_hat[,2])), K =
  RMSE_Multilevel[j] <- rbind(sqrt((1/50)*t(al8$true_mu-EM_out$mu)%*%(al8$true_mu-EM_out$mu)))
}

mmRMSE <- mean(RMSE_Multilevel)
msdRMSE <- sd(RMSE_Multilevel)
```

```r
# naive RMSE (mean of itr=200) al8

lam <- NULL
RMSE_Naive <- NULL

for(j in 1:200){
  GCV_vec <- NULL
  al8 <- mlsplines::generate_response_smooth(J=50, mod="lm", e_sigma=8, N_s=50)

  beta_hat <- NULL
  for(m in 1:50){
    results <- mlsplines::granular(unlist(al8$x_list[m]), unlist(al8$y[m]), mod = "lm")
    beta_hat <- rbind(beta_hat,results)
  }

  K <- mlsplines::make_K(al8$z)

    for(i in 1:length(lambda)){
      naive_out <- mlsplines::naive_ss(beta_hat_vec = unlist(beta_hat[,1]), lambda = lambda[i], K = K)
      GCV_vec <- rbind(GCV_vec,naive_out$GCV)
    }

  lam[j] <- rbind(lambda[which.min(GCV_vec)])
  naive_out <- mlsplines::naive_ss(beta_hat_vec = unlist(beta_hat[,1]), K = K, lam[j])
  RMSE_Naive[j] <- rbind(sqrt((1/50)*t(al8$true_mu-naive_out$mu)%*%(al8$true_mu-naive_out$mu)))
}

nmRMSE <- mean(RMSE_Naive)
nsdRMSE <- sd(RMSE_Naive)

result_mean <- cbind(smRMSE, nmRMSE, mmRMSE)
result_sd <- c(ssdRMSE, nsdRMSE, msdRMSE)
result1 <- data.frame(c(result_mean),c(result_sd))

result1
```

```
##   c.result_mean. c.result_sd.
## 1      0.9208478   0.11012538
## 2      0.3929885   0.11408158
## 3      0.3626143   0.09472641
```

```r
# Single RMSE (mean of itr=200) bl8

lam <- NULL
RMSE_Single <- NULL

for(j in 1:200){
  GCV_vec <- NULL
  bl8 <- mlsplines::generate_response(J=50, mod="lm", e_sigma=8, N_s=50)

  beta_hat <- NULL
  for(m in 1:50){
    results <- mlsplines::granular(unlist(bl8$x_list[m]), unlist(bl8$y[m]), mod = "lm")
    beta_hat <- rbind(beta_hat,results)
```

```
  }

  K <- mlsplines::make_K(bl8$z)

  beta_hat_vec <- unlist(beta_hat[,1])
  RMSE_Single[j] <- rbind(sqrt((1/50)*t(bl8$true_mu- beta_hat_vec)%*%(bl8$true_mu - beta_hat_vec)))
}

smRMSE <- mean(RMSE_Single)
ssdRMSE <- sd(RMSE_Single)


# multilevel RMSE (mean of itr=200) bl8
lambda <- c(10000, 1000, 100, 10, 1, 0.1, 0.01, 0.001, 0.0001)

lam <- NULL
RMSE_Multilevel <- NULL

for(j in 1:200){
  GCV_vec <- NULL
  bl8 <- mlsplines::generate_response(J=50, mod="lm", e_sigma=8, N_s=50)

  beta_hat <- NULL
  for(m in 1:50){
    results <- mlsplines::granular(unlist(bl8$x_list[m]), unlist(bl8$y[m]), mod = "lm")
    beta_hat <- rbind(beta_hat,results)
  }

  K <- mlsplines::make_K(bl8$z)

  for(i in 1:length(lambda)){
    EM_out <- mlsplines::main_EM(beta_hat_vec = unlist(beta_hat[,1]), V = diag(unlist(beta_hat[,2])), K
    GCV_vec <- rbind(GCV_vec,EM_out$GCV)
  }

  lam[j] <- rbind(lambda[which.min(GCV_vec)])
  EM_out <- mlsplines::main_EM(beta_hat_vec = unlist(beta_hat[,1]), V = diag(unlist(beta_hat[,2])), K =
  RMSE_Multilevel[j] <- rbind(sqrt((1/50)*t(bl8$true_mu-EM_out$mu)%*%(bl8$true_mu-EM_out$mu)))
}

mmRMSE <- mean(RMSE_Multilevel)
msdRMSE <- sd(RMSE_Multilevel)

# naive RMSE (mean of itr=200) bl8
lambda <- c(1000000, 100000, 10000, 1000, 100, 10, 1, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0.000001, 0.00

lam <- NULL
RMSE_Naive <- NULL

for(j in 1:200){
  GCV_vec <- NULL
  bl8 <- mlsplines::generate_response(J=50, mod="lm", e_sigma=8, N_s=50)
```

```
  beta_hat <- NULL
  for(m in 1:50){
    results <- mlsplines::granular(unlist(bl8$x_list[m]), unlist(bl8$y[m]), mod = "lm")
    beta_hat <- rbind(beta_hat,results)
  }

  K <- mlsplines::make_K(bl8$z)

  for(i in 1:length(lambda)){
    naive_out <- mlsplines::naive_ss(beta_hat_vec = unlist(beta_hat[,1]), K = K, lambda[i])
    GCV_vec <- rbind(GCV_vec,naive_out$GCV)
  }

  lam[j] <- rbind(lambda[which.min(GCV_vec)])
  naive_out <- mlsplines::naive_ss(beta_hat_vec = unlist(beta_hat[,1]), K = K, lam[j])
  RMSE_Naive[j] <- rbind(sqrt((1/50)*t(bl8$true_mu-naive_out$mu)%*%(bl8$true_mu-naive_out$mu)))
}

nmRMSE <- mean(RMSE_Naive)
nsdRMSE <- sd(RMSE_Naive)

result_mean <- cbind(smRMSE, nmRMSE, mmRMSE)
result_sd <- c(ssdRMSE, nsdRMSE, msdRMSE)
result2 <- data.frame(c(result_mean),c(result_sd))

result2
```

```
##   c.result_mean. c.result_sd.
## 1      0.9268547   0.10248117
## 2      0.7519503   0.10809077
## 3      0.6991136   0.09733331
```