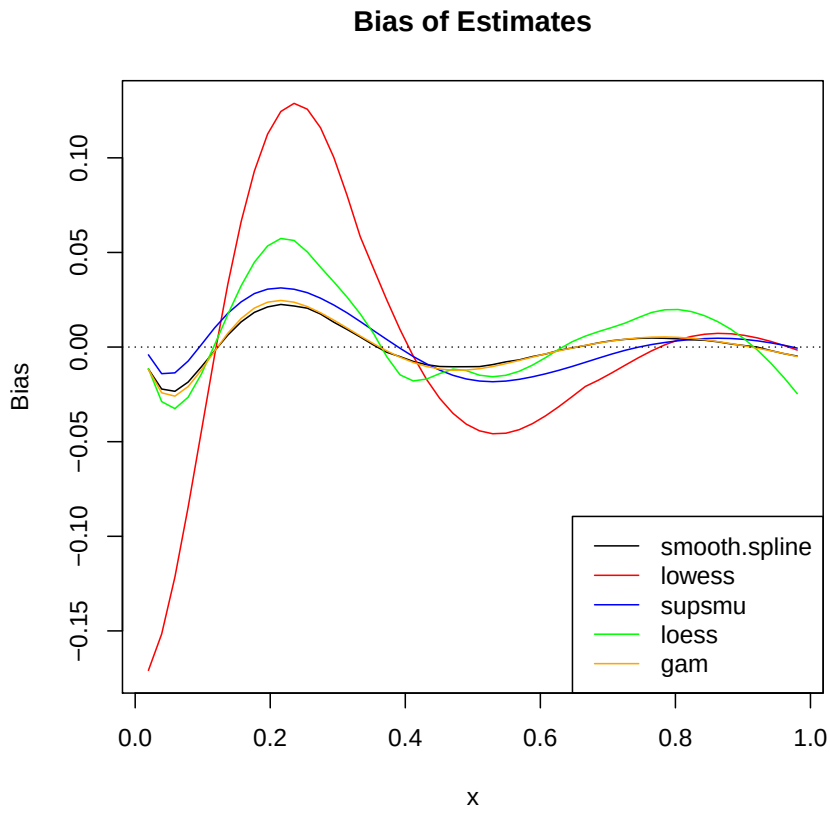
# STAT7400 HW8, 2017

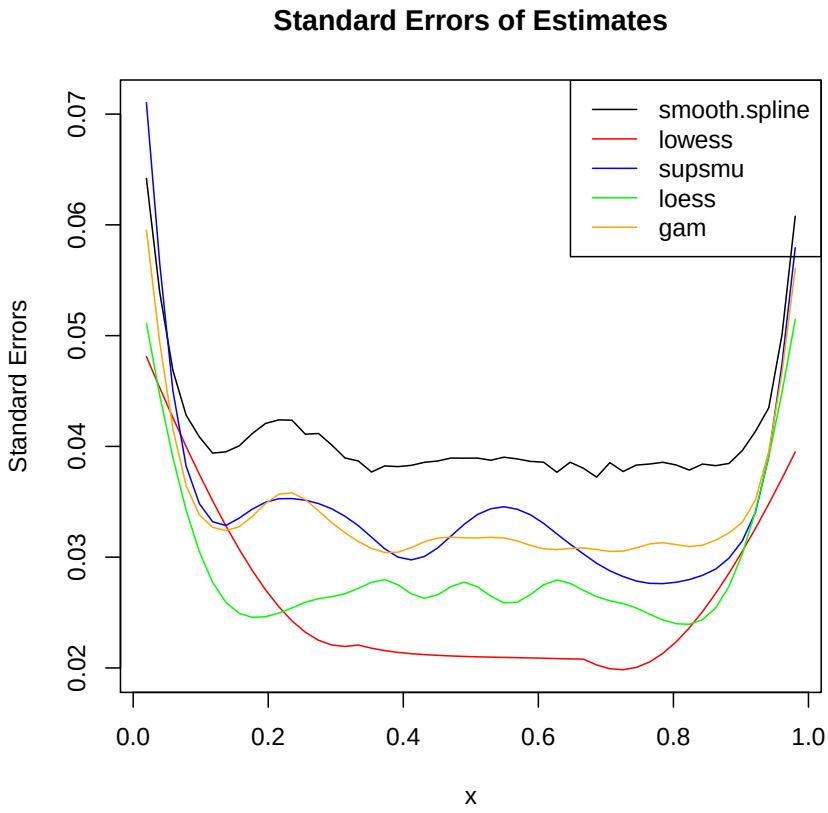
Yiheng Liu

Mar 27, 2017

## Problem 1

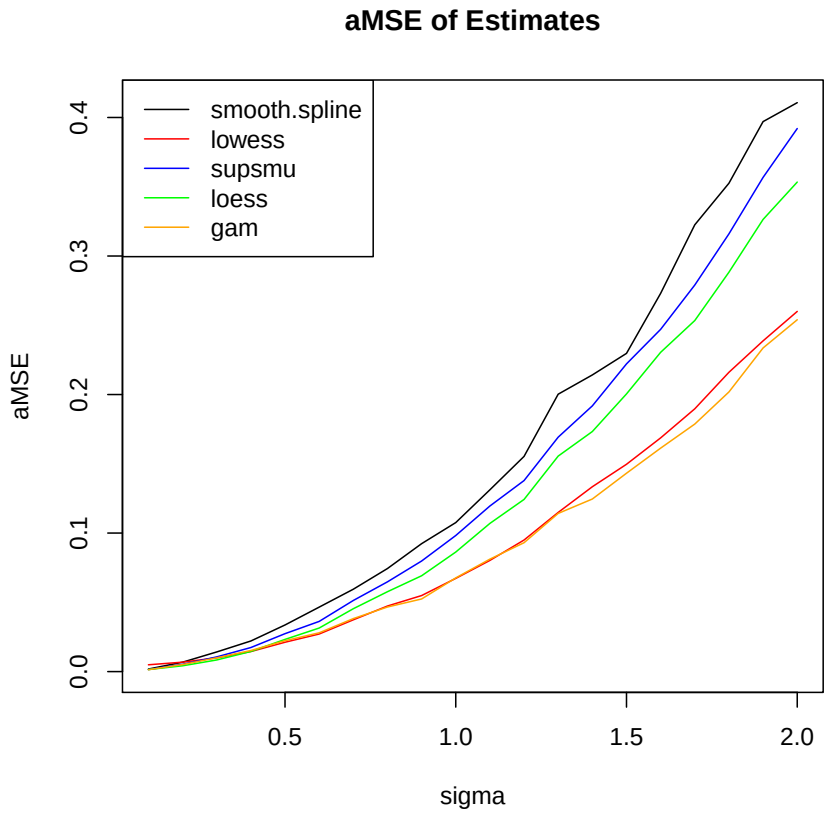
(a) By Monte Carlo simulation of 10000 times, the bias and standard error of the estimates at each $x_i$ are given below.






As we can see, there is a tradeoff between bias and standard error for different methods. For example, `lowess` has a large bias but relatively small standard error. On the contrary, `smooth.spline` has a large standard error but small bias.

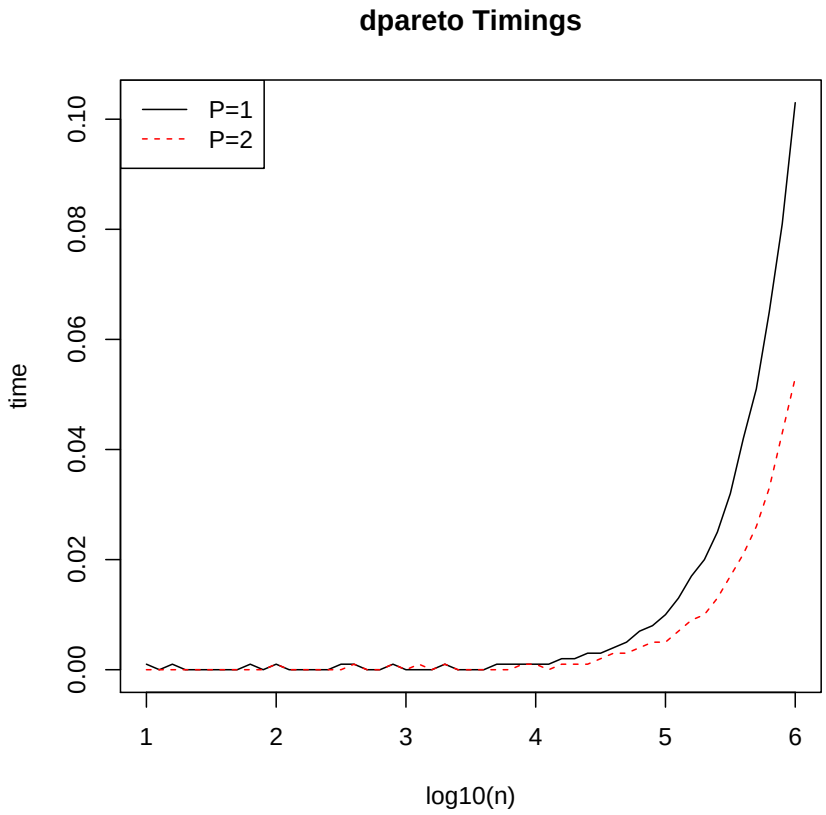
(b) Set the simulation sample size to be 1000 and let $\sigma$ range from 0.1 to 2, the graph of integrated mean square error is given below.



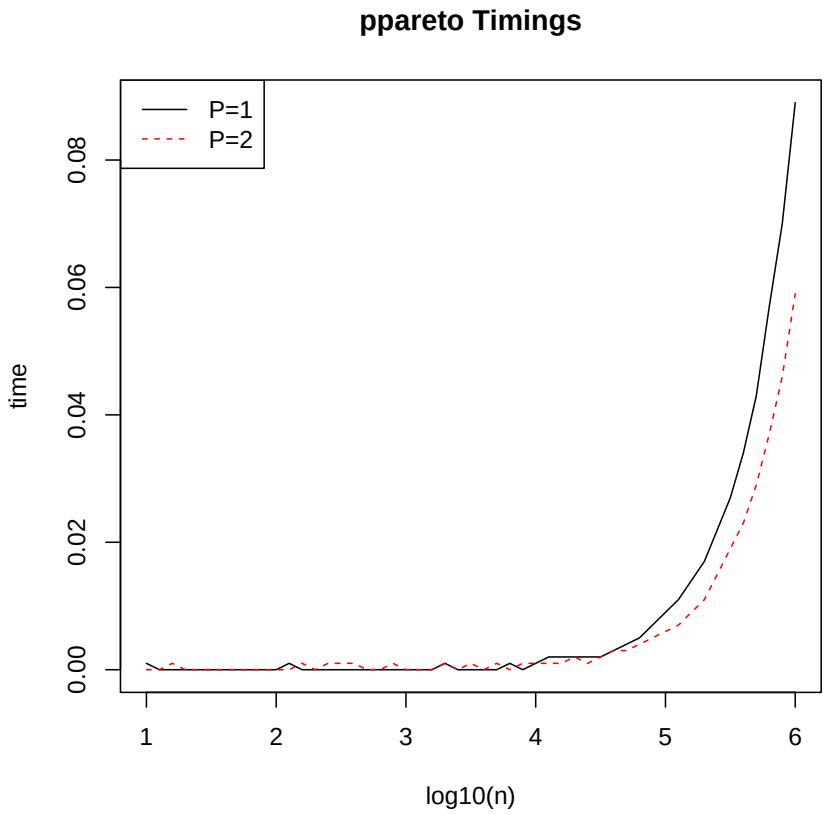

As sigma increases, the integrated mean square error will also increase. For this particular problem, `gam` has the best performance and `smooth.spline` performs the worst in terms of mean square error.

## Problem 2

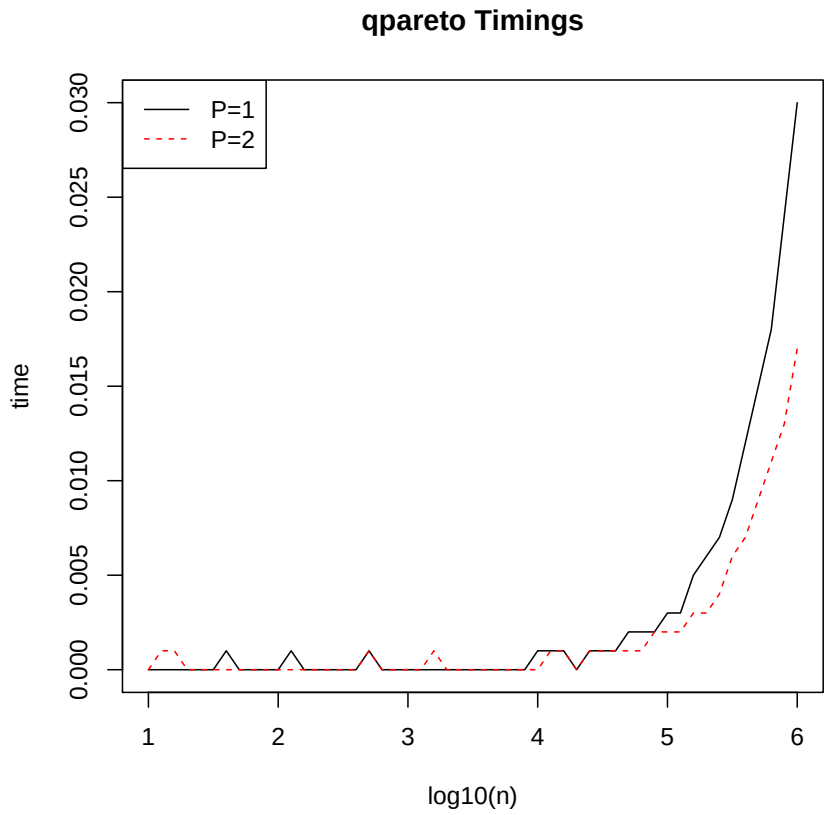
(a) Compare `dpareto` and `p.dpareto`.




(b) Compare `ppareto` and `p.ppareto`.




(c) Compare `qpareto` and `p.qpareto`.




As we can see from the above three graphs, vector has to be more than 100000 for the parallel approach to improve performance.

# Appendix

```
# Problem 1
#(a)
library(mgcv)
n <- 50
sigma <- 0.1
N <- 10^4
i <- c(1:n)
x <- i/(n + 1)

m <- function(x) {
    1 - sin(5 * x)^2 * exp(-4 * x)
}
```

```
m.x <- m(x)

# Matrices to store the fitted values
fit.sm <- matrix(NA, N, n)
fit.low <- matrix(NA, N, n)
fit.sup <- matrix(NA, N, n)
fit.loe <- matrix(NA, N, n)
fit.gam <- matrix(NA, N, n)

set.seed(100)
for (j in 1:N) {
    eps <- rnorm(n, 0, sigma)
    y <- m.x + eps
    fit.sm[j, ] <- smooth.spline(x, y)$y
    fit.low[j, ] <- lowess(x, y)$y
    fit.sup[j, ] <- supsmu(x, y)$y
    fit.loe[j, ] <- loess(y ~ x)$fitted
    fit.gam[j, ] <- gam(y ~ s(x))$fitted
}

# Calculate bias
m.true <- matrix(m.x, N, n, byrow = TRUE)
fit.sm.bias <- colMeans(fit.sm - m.true)
fit.low.bias <- colMeans(fit.low - m.true)
fit.sup.bias <- colMeans(fit.sup - m.true)
fit.loe.bias <- colMeans(fit.loe - m.true)
fit.gam.bias <- colMeans(fit.gam - m.true)

# Plot bias
plot(x, fit.sm.bias, ylab = "Bias", main = "Bias of Estimates",
     ylim = range(c(fit.sm.bias, fit.low.bias, fit.sup.bias, fit.loe.bias,
                    fit.gam.bias)), type = "l")
lines(x, fit.low.bias, col = "red")
lines(x, fit.sup.bias, col = "blue")
lines(x, fit.loe.bias, col = "green")
lines(x, fit.gam.bias, col = "orange")
abline(0, 0, lty = 3)
legend("bottomright", legend = c("smooth.spline", "lowess", "supsmu", "loess",
                                 "gam"), lty = 1,
       col = c("black", "red", "blue", "green", "orange"))

# Calcualte standard error
fit.sm.se <- apply(fit.sm, 2, sd)
fit.low.se <- apply(fit.low, 2, sd)
fit.sup.se <- apply(fit.sup, 2, sd)
```

```
fit.loe.se <- apply(fit.loe, 2, sd)
fit.gam.se <- apply(fit.gam, 2, sd)

# Plot standard error
plot(x, fit.sm.se, ylab = "Standard Errors",
     main = "Standard Errors of Estimates",
     ylim = range(c(fit.sm.se, fit.low.se, fit.sup.se, fit.loe.se, fit.gam.se)),
     type = "l")
lines(x, fit.low.se, col = "red")
lines(x, fit.sup.se, col = "blue")
lines(x, fit.loe.se, col = "green")
lines(x, fit.gam.se, col = "orange")
legend("topright", legend = c("smooth.spline", "lowess", "supsmu", "loess",
                              "gam"), lty = 1,
       col = c("black", "red", "blue", "green", "orange"))

# (b)
sigma <- seq(0.1, 2, 0.1)
N <- 10^3
fit.sm.amse <- rep(NA, length(sigma))
fit.low.amse <- rep(NA, length(sigma))
fit.sup.amse <- rep(NA, length(sigma))
fit.loe.amse <- rep(NA, length(sigma))
fit.gam.amse <- rep(NA, length(sigma))

# Matrices to store the fitted values
fit.sm <- matrix(NA, N, n)
fit.low <- matrix(NA, N, n)
fit.sup <- matrix(NA, N, n)
fit.loe <- matrix(NA, N, n)
fit.gam <- matrix(NA, N, n)
m.true <- matrix(m.x, N, n, byrow = TRUE)

# Calcualte aMSE
set.seed(100)
for (i in 1:length(sigma)) {
    for (j in 1:N) {
        eps <- rnorm(n, 0, sigma[i])
        y <- m.x + eps
        fit.sm[j, ] <- smooth.spline(x, y)$y
        fit.low[j, ] <- lowess(x, y)$y
        fit.sup[j, ] <- supsmu(x, y)$y
        fit.loe[j, ] <- loess(y ~ x)$fitted
        fit.gam[j, ] <- gam(y ~ s(x))$fitted
    }
```

```
    fit.sm.amse[i] <- mean(colMeans((fit.sm - m.true)^2))
    fit.low.amse[i] <- mean(colMeans((fit.low - m.true)^2))
    fit.sup.amse[i] <- mean(colMeans((fit.sup - m.true)^2))
    fit.loe.amse[i] <- mean(colMeans((fit.loe - m.true)^2))
    fit.gam.amse[i] <- mean(colMeans((fit.gam - m.true)^2))
}

# Plot aMSE
plot(sigma, fit.sm.amse, ylab = "aMSE", main = "aMSE of Estimates",
     ylim = range(c(fit.sm.amse, fit.low.amse, fit.sup.amse, fit.loe.amse,
                    fit.gam.amse)), type = "l")
lines(sigma, fit.low.amse, col = "red")
lines(sigma, fit.sup.amse, col = "blue")
lines(sigma, fit.loe.amse, col = "green")
lines(sigma, fit.gam.amse, col = "orange")
legend("topleft", legend = c("smooth.spline", "lowess", "supsmu", "loess",
                             "gam"), lty = 1,
       col = c("black", "red", "blue", "green", "orange"))
```