Verifying and assessing the performance of vsn with simulated data

Wolfgang Huber

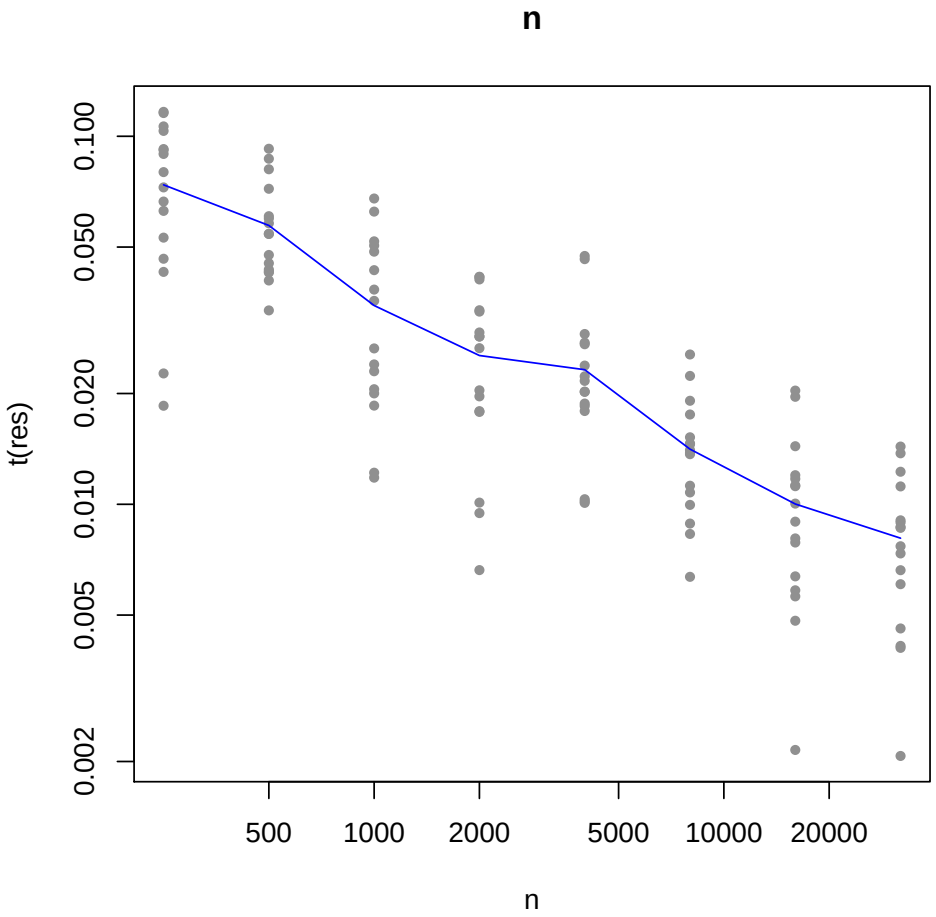
January 28, 2004

There are two functions `sagmbSimulateData` and `sagmbAssess` that can be used to generate simulated data and assess the difference between the 'true' and 'estimated' data calibration and transformation by *vsn*. This vignette demonstrates some examples. Reference [1] describes in more detail (i) the simulation model, (ii) the assessment strategy, and (iii) a comprehensive suite of assessments with respect to the number of probes `n`, the number of arrays `d`, the fraction of differentially expressed genes `de`, and the fraction of up-regulated genes `up`.

```
> library(vsn)
> set.seed(1)
> sim <- function(..., nrrep = 16) {
+     callpar <- list(...)
+     ll <- sapply(callpar, length)
+     stopifnot(ll[1] > 1, all(ll[-1] == 1))
+     res <- matrix(NA, nrow = nrrep, ncol = ll[1])
+     simpar <- append(callpar, list(n = 4096, d = 2, de = 0, up = 0.5,
+         nrstrata = 1))
+     simpar <- simpar[!duplicated(names(simpar))]
+     for (i in 1:ll[1]) {
+         simpar[[1]] <- callpar[[1]][i]
+         for (r in 1:nrrep) {
+             sim <- do.call("sagmbSimulateData", simpar)
+             ny <- vsn(sim$y, strata = sim$strata, verbose = FALSE)
+             res[r, i] <- sagmbAssess(exprs(ny), sim)
+         }
+     }
+     return(res)
+ }
```

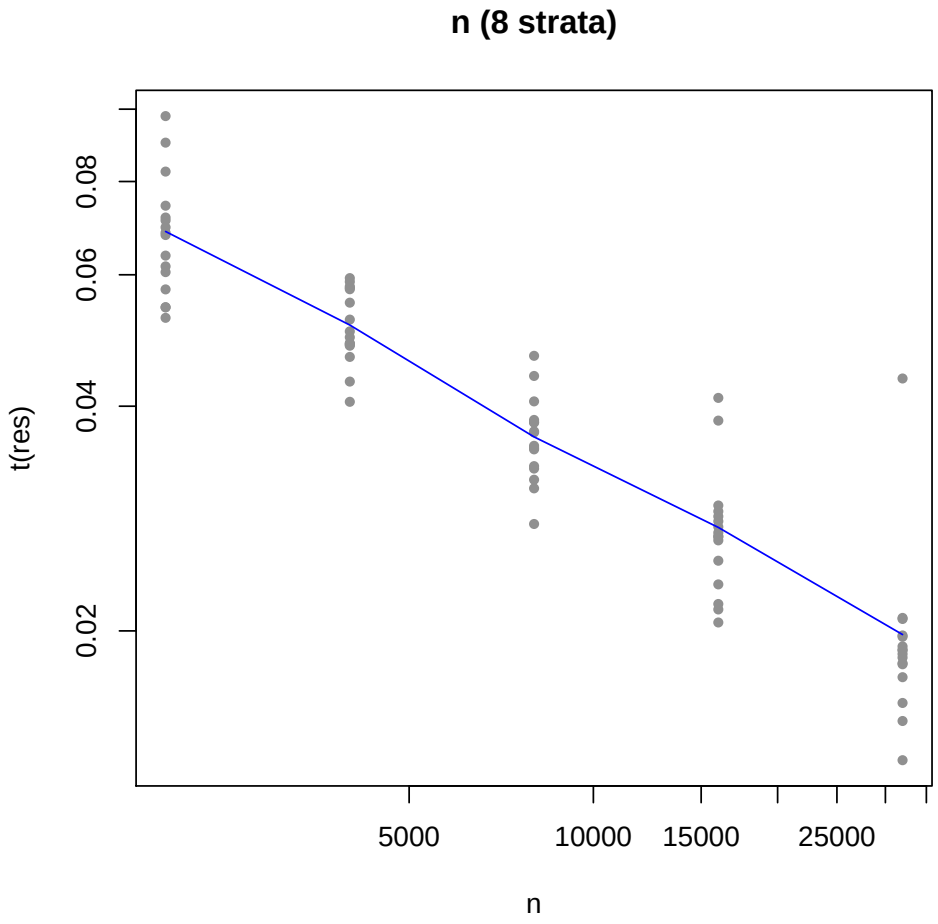
The following plot shows the estimation error for the transformation (i. e. the root mean squared difference between true and estimated transformed data) as a function of the number of genes n. If `vsn` works correctly, the estimation error should decrease roughly as $n^{-1/2}$.

```
> n <- 1000 * 2^seq(-2, 5)
> res <- sim(n = n)
> matplot(n, t(res), pch = 20, log = "xy", col = "#909090", main = "n")
> lines(n, colMeans(res), col = "blue")
```


n
t(res)
0.002
0.005
0.010
0.020
0.050
0.100
500
1000
2000
5000
10000
20000
n

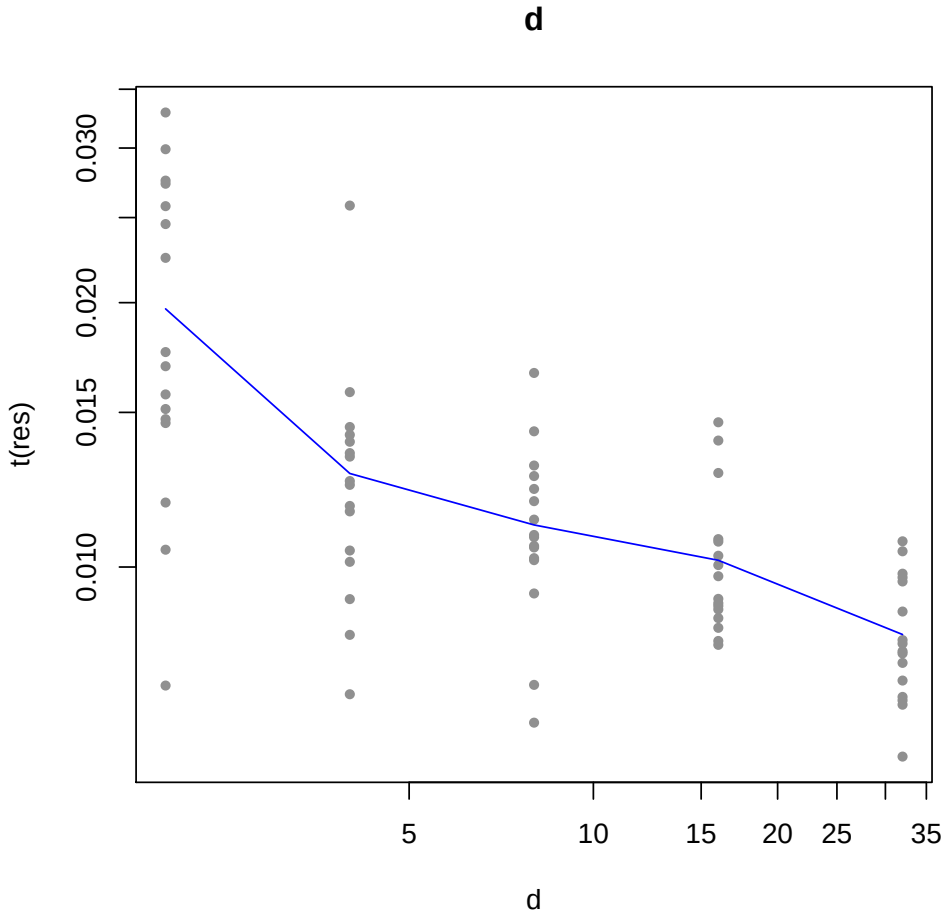
The same with 8 strata:

```
> n <- 1000 * 2^seq(1, 5)
> res <- sim(n = n, nrstrata = 8)
> matplot(n, t(res), pch = 20, log = "xy", col = "#909090", main = "n (8 strata)")
> lines(n, colMeans(res), col = "blue")
```



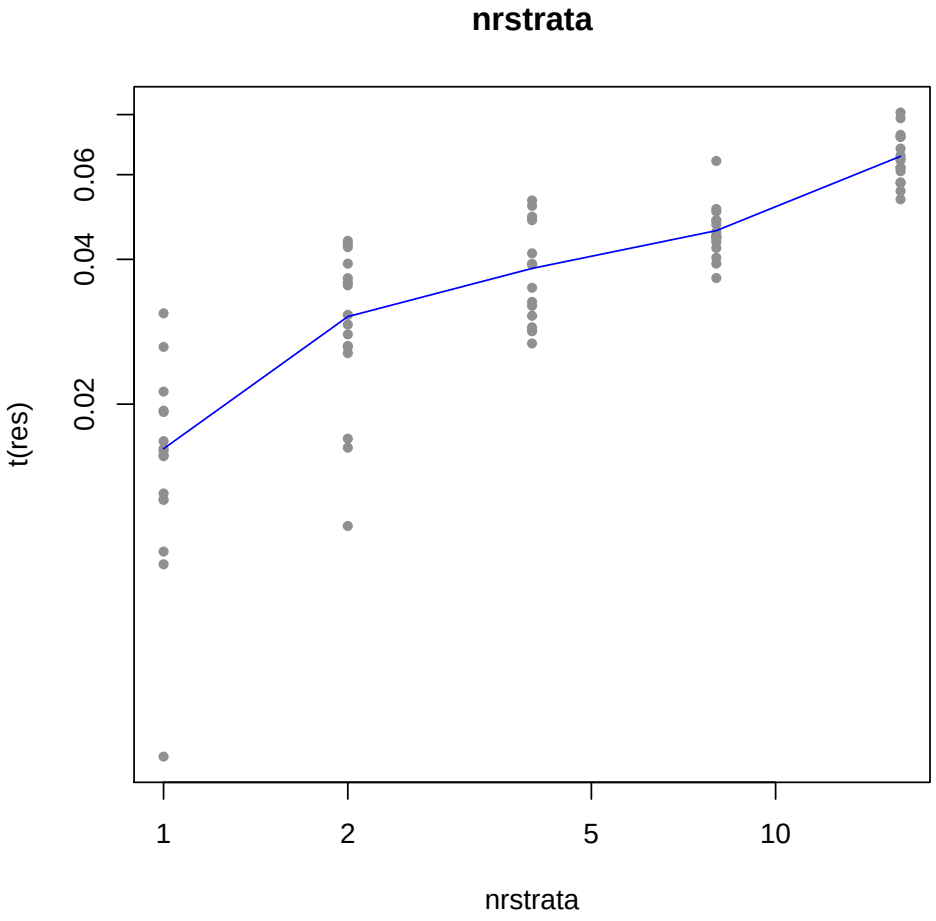
The following plot shows the estimation error as a function of the number of samples d. Initially, it also decreases slightly with d, but eventually reaches a plateau. This is because the number of parameters that need to be estimated is proportional to d, so the "number of data points per parameter" is constant in this plot (above, it was increasing proportionally to n).

```
> d <- 2^seq(1, 5)
> res <- sim(d = d)
> matplot(d, t(res), pch = 20, log = "xy", col = "#909090", main = "d")
> lines(d, colMeans(res), col = "blue")
```



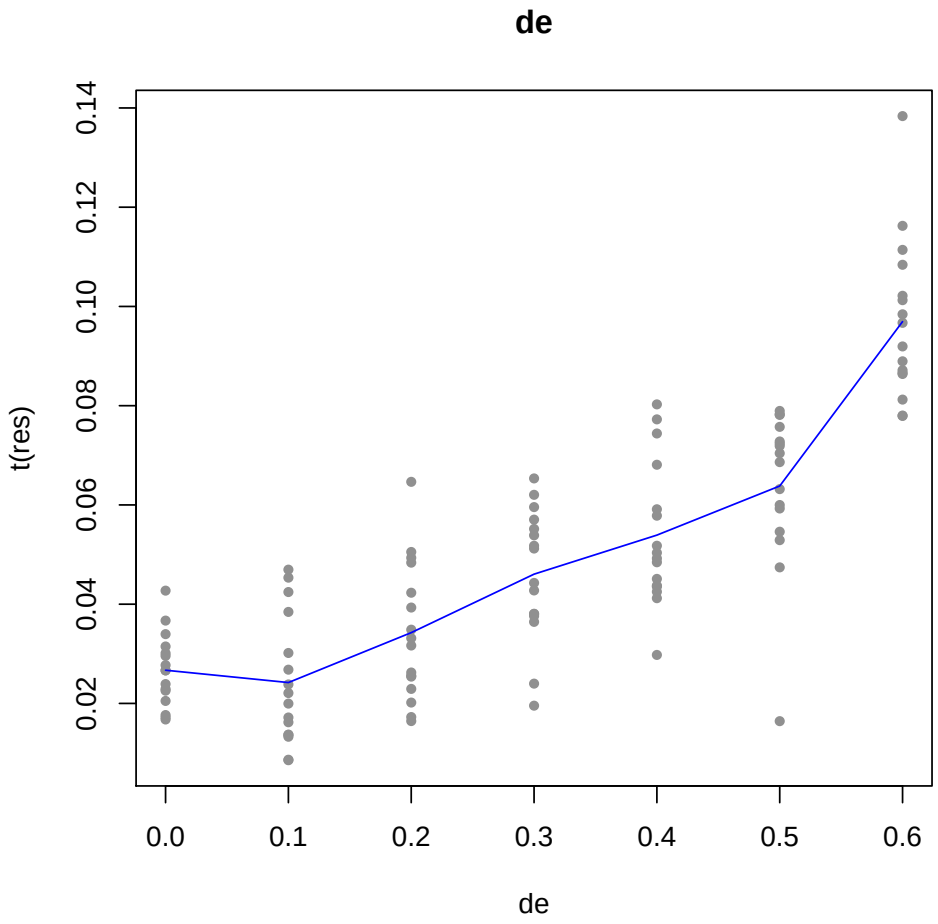
Here, we see the estimation error as a function of the number of strata. It should increase, since for each stratum, we need to estimate separate parameters, and if the overall number of probes does not change, more strata means less and less data per parameters.

```
> nrstrata <- 2^seq(0, 4)
> res <- sim(nrstrata = nrstrata)
> matplot(nrstrata, t(res), pch = 20, log = "xy", col = "#909090",
+     main = "nrstrata")
> lines(nrstrata, colMeans(res), col = "blue")
```



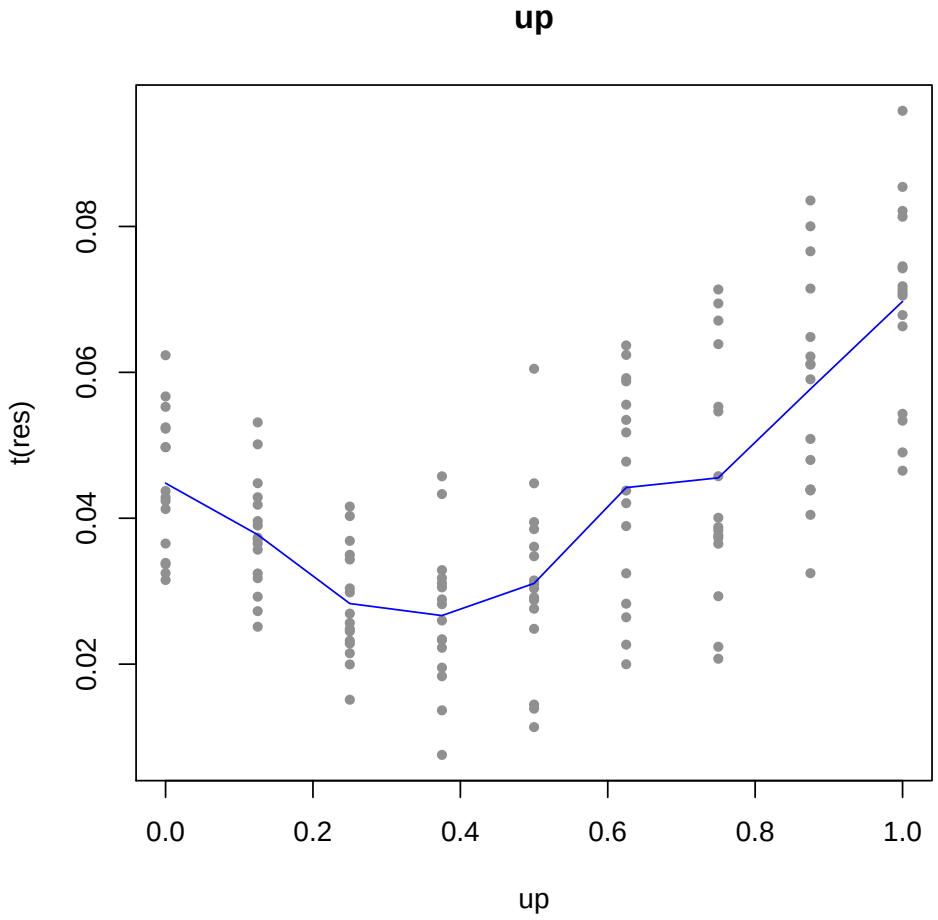
`de` is the fraction of differentially expressed genes.

```
> de <- (0:6)/10
> res <- sim(de = de)
> matplot(de, t(res), pch = 20, col = "#909090", main = "de")
> lines(de, colMeans(res), col = "blue")
```



up is the fraction of up-regulated genes among the differentially expressed genes. The best results are obtained for up$\approx$ 0.5, while the estimation goes up the more unbalanced the situation becomes.

```
> up <- (0:8)/8
> res <- sim(up = up, de = 0.2)
> matplot(up, t(res), pch = 20, col = "#909090", main = "up")
> lines(up, colMeans(res), col = "blue")
```



References

[1] W. Huber, A. von Heydebreck, H. Sültmann, A. Poustka, and M. Vingron. Parameter estimation for the calibration and variance stabilization of microarray data. *Statistical Applications in Genetics and Molecular Biology*, Vol. 2: No. 1, Article 3, 2003. http://www.bepress.com/sagmb/vol2/iss1/art3