R Design Patterns: Base-R vs. Tidyverse
With an eye toward the teaching of R beginners

Norman Matloff
Dept. of Computer Science, University of California, Davis

August 18, 2022

This document enables the reader to see at a glance the difference between base-R and the tidyverse in common R design settings. I believe the base-R versions are generally simpler, thus more appropriate for R learners.

All examples use R's built-in datasets. After e.g., changing a data frame, it is restored for the next example, e.g. **data(mtcars)**. The examples are presented roughly in order of how often these operations tend to be performed by R users.

As this document is aimed at comparing base-R and the tidyverse in terms of teaching new R learners, advanced functions from either base-R or the tidyverse are excluded here.

More and more examples will be added over time.

Reading a specific cell in a data frame

```
mtcars$mpg[3]
```

```
select(mtcars,mpg) %>%
   filter(row_number() == 3)
```

Adding a column to a data frame

```
mtcars$hwratio
  <- mtcars$hp / mtcars$wt
```

```
mutate(mtcars, hwratio=hp/wt) -> mtcars
```

Of course, typically Tidy coders would use <− rather than −>. I feel that the former is more consistent with the “left to right flow” of pipes. But in any case, the point about code complexity is the same either way.

Extracting rows

```
mtc8 <-
   subset(mtcars, cyl==8)
```

```
filter(mtcars,mpg > 15) -> mtc8
```

Mean by group

```
tapply(mtcars$mpg,
   mtcars$cyl,mean)
```

```
group_by(mtcars, cyl) %>%
   summarize(meanMPG =
      mean(mpg,))
```

Quick look at univariate data

```
hist(Nile)
```

```
dn <- data.frame(Nile)
ggplot(dn) +
  geom_histogram(aes(Nile),dn)
```

One could do this more quickly in **qplot()** than in the full **ggplot2**, but this is seldom taught in tidyverse courses.

Quick look at bivariate data

```
plot(x=mtcars$wt,
  y=mtcars$mpg)
```

```
ggplot(data = mtcars,
  aes(x = wt, y = mpg))
  + geom_point()
```

Binary categorization on a vector

```
NileHiLow <-
  ifelse(Nile >= 1000,
    'high','low')
```

```
Nile %>% as.data.frame %>%
  mutate(
    HighLow = case_when
    (x < 1000~'low',
     x >= 1000~'high')
  ) %>%
  select(HighLow) %>%
  as.vector -> HighLow
```

The step of conversion back to a vector at the end is needed for many R packages in which vector input is required.

Deleting columns from a data frame

```
mtcars[c('drat','carb')]
  <- NULL
```

```
select(mtcars,-c(drat,carb))
  -> mtcars
```