Quick tricks for sequential string or character names.

Dr. Jon Starkweather, Research and Statistical Support consultant.

This month's article is just a short piece which may be very useful when working with large data sets. The article offers tips for naming objects which contain a large number of elements. The primary reason for this article is the ability to create a sequential character string. Often this is handy when trying to create a sequence of names for columns or rows of a matrix or data frame and the number of names (or labels) is so large as to make typing them manually quite time consuming -- the script below automates the process in a some-what generic way which can be applied to a variety of situations. The examples below are very small but, allow illustration of how these techniques would be applied to a large, or very large, data situation. The primary **R** function involved is the 'paste' function, which is available in the base package (which is included with the initial installation of R).

Example 1: Creating a vector of sequential names.

A vector of names can easily be applied to the columns of a matrix or data frame. Let's say we have 20 survey questions, or items, and we want the names of the items to be sequential so they reflect their order in which the respondents were exposed to them. First, set the number of objects we are going to name. In this example we have 20.

```
n <- 20
```

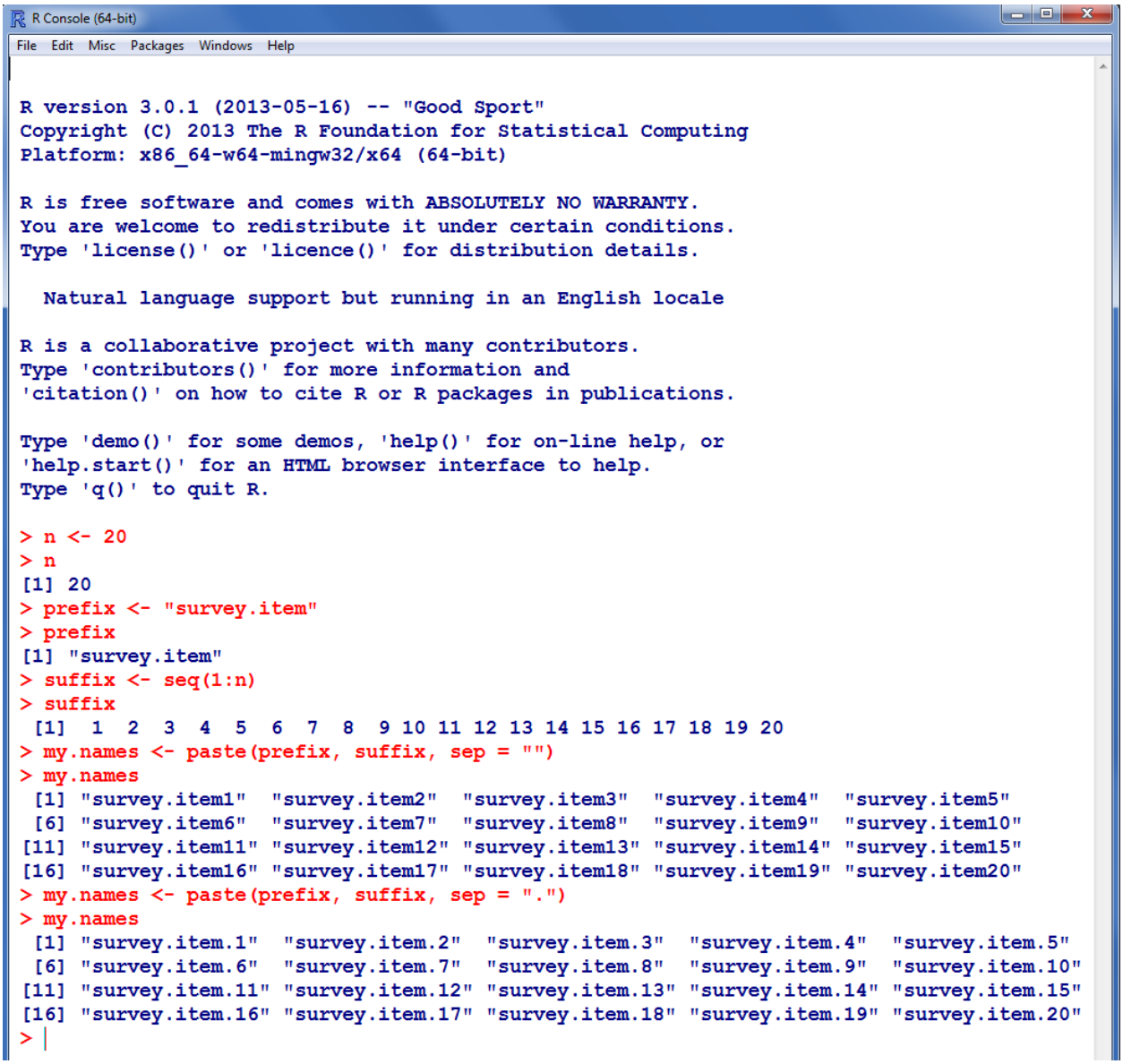
Second, create the prefix of the names.

```
prefix <- "survey.item"
```

Third, create a sequence (vector) of values to be the suffix. The suffix will be attached or *pasted* to the prefix to form the sequential names.

```
suffix <- seq(1:n)
```

Fourth, create a vector which takes the prefix and attaches the suffix as a character string. Note; there are two examples below. The first contains no separator (`sep = ""`) between the prefix and suffix; the second example contains a period as the separator (`sep = "."`).



Above, we can see how the name stem, or prefix, gets attached to each element of the sequential vector, or suffix, to create the sequentially numbered names; which themselves are character strings. These names could then be applied to the columns of a data set (matrix or data.frame) using the typical '*names*' function.

```
names(my.data) <- my.names
```

Example 2: Creating a matrix of character string elements.

In this example, we are creating a matrix of names identifying the internal cells of the matrix sequentially. First, how many rows and columns (or cells) are you trying to create and name? Here, 10 rows by 5 columns; 50 cells.

```
n.rows <- 10
n.cols <- 5
```

```
n.e <- n.rows * n.cols
```

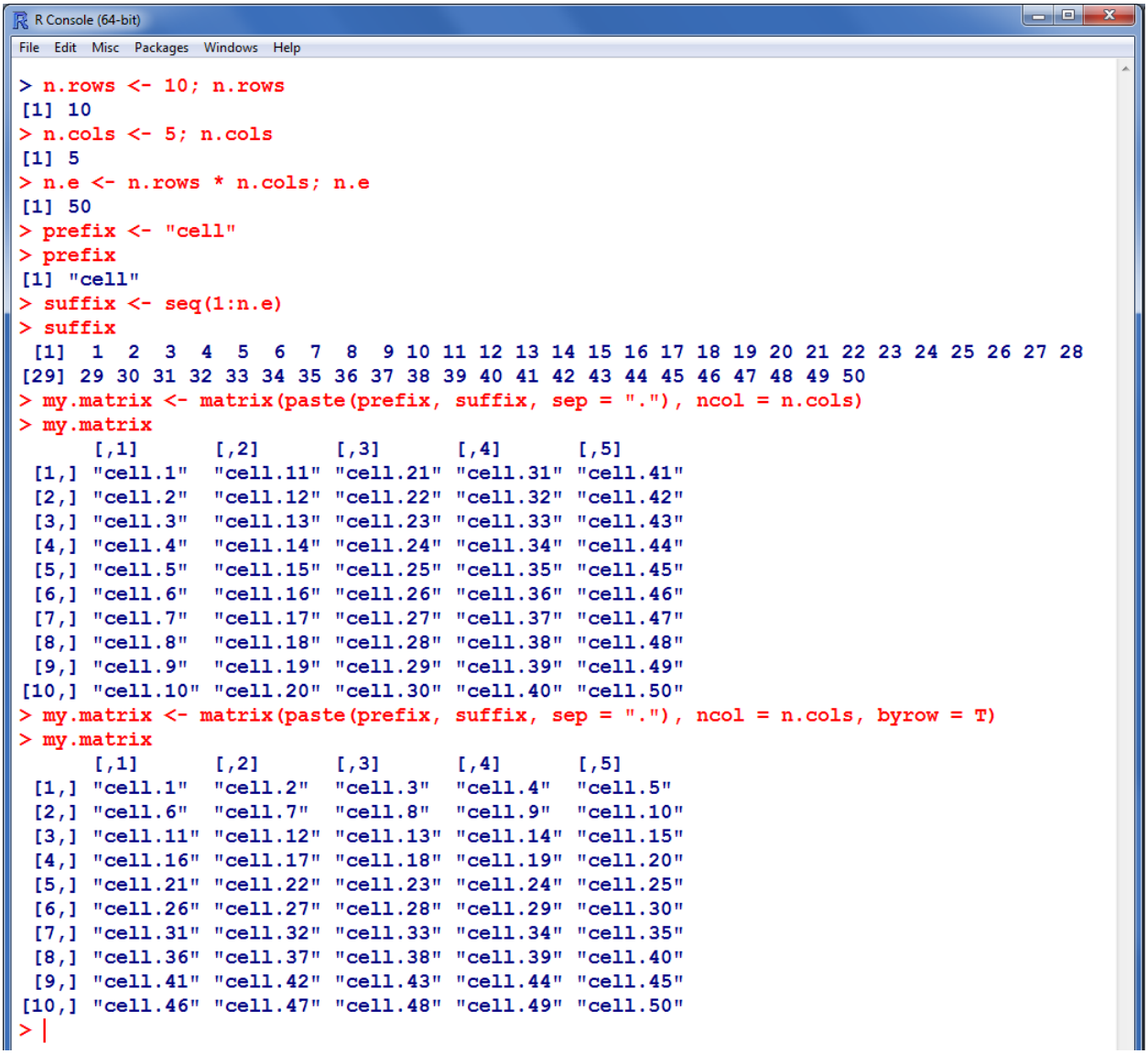
Next, create the prefix character string, here we are sampling using ‘cell’ as the prefix.

```
prefix <- "cell"
```

Next, create the sequence suffix.

```
suffix <- seq(1:n.e)
```

Next, combine prefix and suffix while creating a matrix; first, with the sequence ordered down then across. You can also order the sequence across then down using the ‘`byrow = TRUE`’ argument, as is shown in the second matrix below.



Example 3: Creating a matrix with 'row by column' identifiers.

In this example, we create a matrix in which each internal cell is identified sequentially by its row and column location within the matrix. Again, we need to setup our matrix first by specifying the number of rows and columns (or cells) we are trying to create.

```
n.rows <- 10
n.cols <- 5
n.e <- n.rows * n.cols
```

Next, create the prefix character string.

```
prefix <- "cell"
```

Next, create the 'row suffix' and 'column suffix' by using the sequential function.

```
r.suffix <- seq(1:n.rows)
c.suffix <- seq(1:n.cols)
```

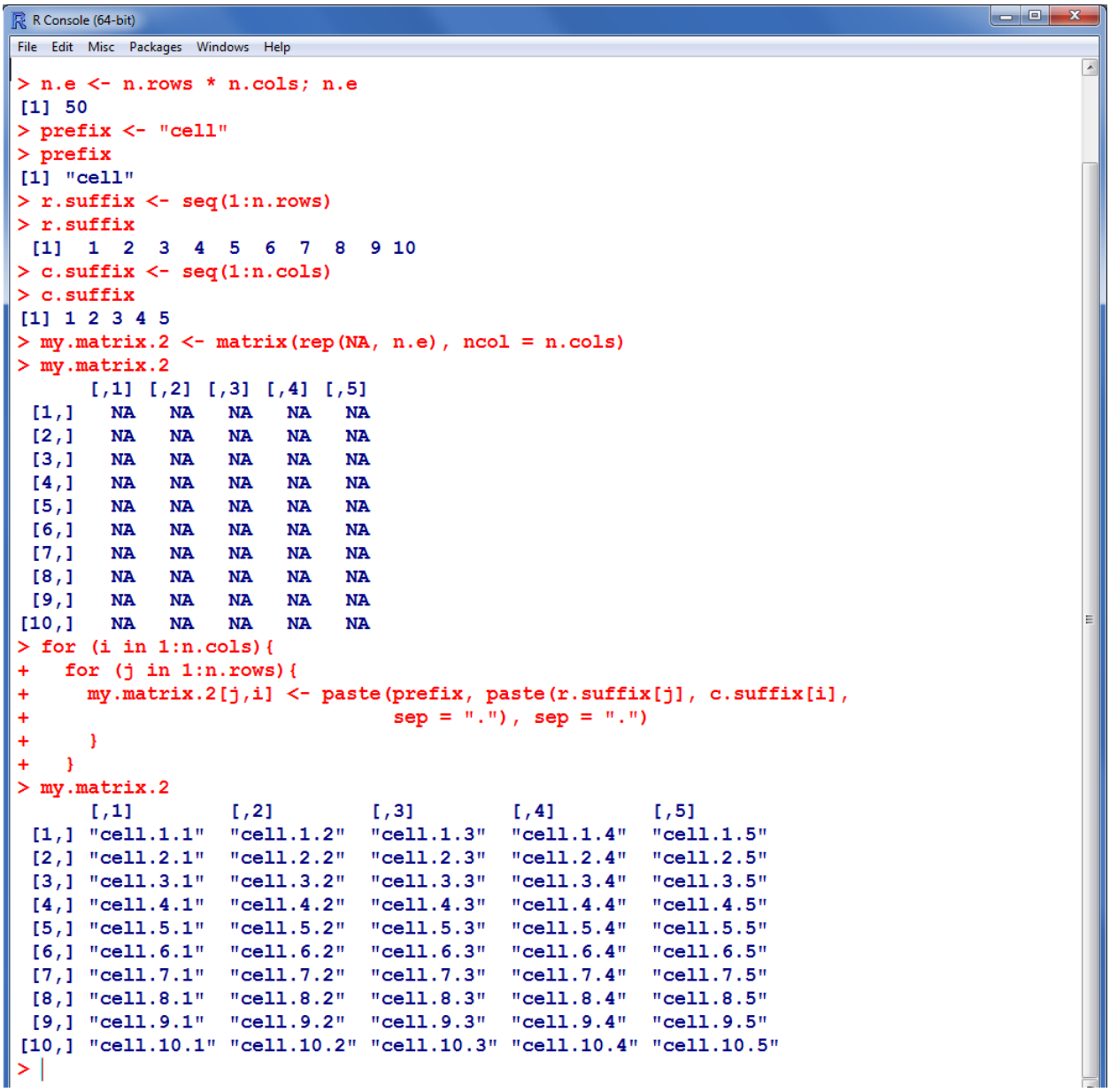
Next, create an empty matrix (each cell is empty: 'NA') in which the sequential character strings will go.

```
my.matrix.2 <- matrix(rep(NA, n.e), ncol = n.cols)
my.matrix.2
```

Next, create an iterative 'for-loop' to combine the elements and fill in the matrix.

```
for (i in 1:n.cols){
  for (j in 1:n.rows){
    my.matrix.2[j,i] <- paste(prefix, paste(r.suffix[j], c.suffix[i],
                                sep = "."), sep = ".")
    }
  }; rm(i,j)

my.matrix.2
```



Again, this article is not meant to be terribly technical; it just presents some handy methods for assigning sequential order to character strings. In the small contrived examples above, this does not seem very useful; but, if the situation involves a large data set with perhaps over 10000 columns and / or perhaps over 100000 rows…then the practical utility of this article will be readily apparent. An R script file with the same information as contained in this article is available at the Research and Statistical Support Do-It-Yourself Introduction to R course website.

Until next time, *happy computing*…