Vector Calculations in R

The following is a direct quote from:


Note that references to S equally apply to R in this context.

Vectorised calculations

User coming to S from other languages are often slow to take advantage of the power of S to do vectorised calculations, that is, calculations that operate on entire vectors rather than on individual components in sequence. This often leads to unnecessary loops. For example, consider calculating the Pearson chi-squared statistics for testing independence in a two-way contingency table. This is defined as

\[ X^2_p = \sum_{i=1}^{r} \sum_{j=1}^{s} \frac{(f_{ij} - e_{ij})^2}{e_{ij}}, \quad e_{ij} = \frac{f_i f_j}{f..} \]

Two nested for loops may seem to be necessary, but in fact no explicit loops are needed. If the frequencies \( f_{ij} \) are held in a matrix the most efficient calculation in S uses matrix operations:

\[
\begin{align*}
fi. &\leftarrow f \ %% \ rep(1, \ ncol(f)) \\
f.j &\leftarrow rep(1, \ nrow(f)) \ %% \ f \\
\text{e} &\leftarrow (fi. \ %% \ f.j)/\text{sum(fi.)} \\
X2p &\leftarrow \text{sum}((f - e)^2 / e)
\end{align*}
\]

Explicit loops in S should be regarded as potentially expensive in time and memory use and ways of avoiding them should be considered. (Note that this will be impossible with genuinely iterative calculations such as our Newton scheme on page 59.)
Example

The following question was posed by Nigel, and my solution in R is attached, with output. I did not attempt a loop solution, although it would be feasible, but inefficient for long vectors.

The posed question:

Hi Bernard
I have a question that i was hoping you could help me with.
It is to do with R.

I have two vectors in the same style as
x<-c(NA,NA,1,3,4,NA,1)
y<-c(0,0,NA,NA,NA,0,NA)

that is when x[ i ] = NA, y[ i ]=numeric.
I wish to replace the NA’s in x by the corresponding y value.
So the end result will be a vector that only has values.
Any help would be much appreciated

Regards,
Nigel
My response (R command line file):

```r
x <- c(NA,NA,1,3,4,NA,1)
y <- c(0,0,NA,NA,NA,0,NA)

cbind(x,y)

z<-y[!is.na(y)]

x[is.na(x)] <- z

# general case

a <-c(10,20,NA,NA,NA,30,NA)

X <- c(NA,NA,1,3,4,NA,1)
w <- a[!is.na(a)]

w

X[is.na(X)] <- w

X
```
The corresponding output:

```r
> x <- c(NA,NA,1,3,4,NA,1)
> y <- c(0,0,NA,NA,NA,0,NA)
>
> cbind(x,y)
   x y
[1,] NA 0
[2,] NA 0
[3,] 1 NA
[4,] 3 NA
[5,] 4 NA
[6,] NA 0
[7,] 1 NA
> z<-y[!is.na(y)]
> z
[1] 0 0 0

> x[is.na(x)] <- z
> x
[1] 0 0 1 3 4 0 1
>
> # general case
>
> a <-c(10,20,NA,NA,NA,30,NA)
> X <- c(NA,NA,1,3,4,NA,1)
> w <- a[!is.na(a)]
> w
[1] 10 20 30
> X[is.na(X)] <- w
> X
[1] 10 20 1 3 4 30 1
```

Bernard A. Ellem