An empty exercise

Carl de Boor

MATLAB is a most convenient, versatile and helpful interactive program for experimental scientific calculations, even on PCs and particularly because of its graphics capabilities. It makes high-quality algorithms for the solution of standard problems in Linear Algebra available to the casual user. It even has an empty matrix (which is more than can be said of most textbooks in Linear Algebra or Matrix Theory or of most programming languages, with APL a notable exception; see, e.g., [BJ]).

The empty matrix provided by MATLAB is the 0×0-matrix []. The MATLAB tutorial [M] gives two uses of [], points out that certain MATLAB matrix functions have been given 'mathematically plausible values' at [], and finishes with the statement: 'As far as we know, the literature on the algebra of empty matrices is itself empty. We're not sure we've done it correctly, or even consistently, but we have found the idea useful.'

This note is intended to point out that empty matrices occur naturally, and that their treatment need not be merely mathematically plausible, but is in fact entirely determined by standard linear algebra considerations. In particular, it is necessary to allow for empty matrices of various sizes $0 \times n$ and $m \times 0$ in addition to [], not only because that is what the theory provides but because these various empty matrices are useful in the same way the empty sum or the empty product is useful, viz. as a convenient and natural way to start inductions.

While it is customary to use various graphic terms, such as 'rectangular array', or 'rectangular arrangement' and the like, to describe a matrix, it seems cleanest to stick to the definition that an $m \times n$-matrix A is a scalar-valued map from the set $\mathbf{m} \times \mathbf{n}$ to the scalar-field $\mathbb{F}$. Here,

$$\mathbf{k} := \{1, 2, \dots, k\}$$

is the set consisting of the first k natural numbers, and the scalar-field $\mathbb{F}$ is just that, a field, typically $\mathbb{F} = \mathbb{R}$ or $\mathbb{C}$. Bourbaki [B;p.73] seems to prefer to define a matrix as the 'indexed family' $(a_{ij})_{(i,j)\in\mathbf{m}\times\mathbf{n}}$, but I fail to appreciate the distinction. Bourbaki defines an 'empty' matrix, but only one such, viz. any matrix obtained when, in this definition, m or n is zero.

I find it easiest to discuss empty matrices in terms of the linear maps which they represent in standard fashion. Recall the n-dimensional coordinate space

$$\mathbb{F}^n := \{a : \mathbf{n} \to \mathbb{F}\},$$

i.e., the set of all n-sequences in $\mathbb{F}$, or, equivalently, the set of all scalar-valued maps on $\mathbf{n}$, with the vector operation provided by pointwise addition and pointwise multiplication by a scalar. Recall further that the matrix $A \in \mathbb{F}^{m\times n}$ is usually identified with the linear map

$$\mathbb{F}^n \to \mathbb{F}^m : a \mapsto \sum_j A(:, j)a(j)$$

which associates with each $a \in \mathbb{F}^n$ the particular linear combination $\sum_j A(:, j)a(j)$ of the columns $A(:, j)$ of A with weight vector a. Recall, finally, that the resulting map from $\mathbb{F}^{m\times n}$ to the linear space $L(\mathbb{F}^n, \mathbb{F}^m)$ of linear maps from $\mathbb{F}^n$ to $\mathbb{F}^m$ is linear and invertible, with the inverse associating each $A \in L(\mathbb{F}^n, \mathbb{F}^m)$ with the matrix $A \in \mathbb{F}^{m\times n}$ whose jth column $A(:,j)$ is given by Ae_j, with e_j the jth unit vector.

Consider now, in particular, the coordinate space $\mathbb{F}^0$. Its sole element is the empty sequence in $\mathbb{F}$, which therefore must be its 0 element. It follows that, for any vector space X, there is exactly one linear map $\mathbb{F}^0 \to X$ and exactly one linear map $X \to \mathbb{F}^0$, viz. the 0 map, i.e., the map which carries its entire domain to the 0 of its target. Correspondingly, $\mathbb{F}^{m\times 0}$ and $\mathbb{F}^{0\times n}$ each consist of exactly one element. The unique $m \times 0$-matrix has no columns, but m rows, while each of the n columns of the unique $0 \times n$-matrix contains the sole element of $\mathbb{F}^0$. Inasmuch as we can also think of $A \in \mathbb{F}^{m\times n}$ as acting on $\mathbb{F}^n$ by carrying the n-vector a to the m-vector $(\langle A(i,:), a\rangle)_{i\in\mathbf{m}}$ (with the rows $A(i,:)$ of A acting as linear functionals on $\mathbb{F}^n$), we can also describe the unique $m \times 0$-matrix as having the unique element of $(\mathbb{F}^0)'$ in each of its m rows.

Since the algebra of matrices is set up to mirror the algebra of linear maps, it is now clear how to include empty matrices in matrix algebra. Since $\mathbb{F}^{m\times n}$ consists of just one element in case m or n is zero, any linear combination of such empty matrices is again that same empty matrix. The product AB of $A \in \mathbb{F}^{m\times n}$ with $B \in \mathbb{F}^{n\times p}$ continues to make good sense even if one (or more) of m, n, or p is zero. AB is always an element of $\mathbb{F}^{m\times p}$, hence the unique element of that set in case m or p is zero. Only the case $n = 0$ might require a moment's thought. In this case, the domain of A is $\mathbb{F}^0$, hence A is the sole linear map from $\mathbb{F}^0$ to $\mathbb{F}^m$, i.e., $\operatorname{ran} A = \{0\}$. Consequently, AB is the zero matrix of size $m \times p$.

Further, any norm of an empty matrix is zero, as the supremum of the empty set of nonnegative numbers. This implies that the condition number of the square empty matrix $[]$ is 0. This makes $[]$ the only matrix with a condition number less than 1 and the only invertible matrix with zero norm. To be sure, $[]$ is indeed invertible, since it is (or represents) the identity id_0 on $\mathbb{F}^0$. More than that, $p([]) = [] = \mathrm{id}_0$ for any polynomial p, hence for any function p. This implies that the spectrum of $[]$ is empty and that its characteristic polynomial equals its minimal annihilating polynomial, viz. the polynomial $()^0 : t \mapsto 1$. Consistent with this (and other considerations) is the choice $\det([]) := 1$.

Empty matrices are useful when considering bases. While the standard definition describes a basis for the vector space X as a linearly independent and spanning sequence $v_1, \ldots, v_n$ in X, it seems more to the point to define a basis as an invertible linear map $V = [v_1, \ldots, v_n]$ from some coordinate space $\mathbb{F}^n$ to X, with $[v_1, \ldots, v_n]$ the map given by the rule

$$[v_1, \ldots, v_n] : \mathbb{F}^n \to X : a \mapsto \sum_j v_j a(j),$$

and then call such n the dimension of X. Textbooks are somewhat tentative when it comes to discussing a basis for the trivial vector space. Some will say that the trivial vector space has no basis, which seems strange since the very same books will define the dimension of a vector space to be the cardinality of a(ny) basis for it, hence would have to conclude that the trivial vector space has no dimension. It would be better if these books would join the rest in saying that the trivial vector space has the empty sequence as a basis, hence is zero-dimensional. The very same conclusion is reached more readily in terms of the definition of a basis advocated here, since $\mathbb{F}^0$ is the only coordinate space $\mathbb{F}^n$ providing an invertible linear from $\mathbb{F}^n$ to the trivial vector space. In this sense, the $m \times 0$-matrix is the unique basis of the trivial subspace of $\mathbb{F}^m$.

There are good reasons for allowing empty matrices of all sizes in matrix-oriented languages such as MATLAB, for the same reason that mathematics has found it convenient to allow empty sums and products: it makes it possible to start off inductive processes with ease. Here are two examples.

Example 1: Extraction of a basis from a spanning sequence. Given a spanning sequence for the subspace X of $\mathbb{F}^m$, i.e., given a $W \in \mathbb{F}^{m\times n}$ with $\operatorname{ran} W = X$, a basis $V \in \mathbb{F}^{m\times r}$ for X is obtained by the following standard algorithm, described here in ideal MATLAB:

```
V=zeros(m,0);
for j=1:n,
   w=W(:,j);
   if w~=V*(V\w), V=[V,w];end
end
```

in which a column `w` of W is adjoined to the 1-1 linear map V (made up from the columns of W already examined and selected) if it is found not to be already in the span or range of V. Here, `V\w` provides the best least-squares solution to the equation `V?  = w`, hence `w` $\in \operatorname{ran} V$ if and only if `w=V*(V\w)`. It is most convenient to initialize V as the $m \times 0$-matrix, since this would avoid having to make special provisions for the case that W is the zero matrix. In fact, the algorithm would also work well if $n = 0$, giving, in either case, the correct conclusion that the dimension of $\operatorname{ran} W$, i.e., the number of columns in a basis for $\operatorname{ran} W$, is zero. (There is no claim here that this is the most efficient way, in this context, to ascertain whether or not `w` $\in \operatorname{ran} V$, nor is there any discussion of the effects of rounding errors on this algorithm. In fact, Gauss elimination, in the guise of the algorithm for reducing W to row-echelon form, would be the efficient way.)

Example 2: Crout-Dolittle. This example is due to Warren Ferguson of SMU (and quoted with his permission): In the Crout algorithm or the Dolittle algorithm for the calculation of an LU factorization, one alternates between computing a column of the lower triangular factor L and computing a row of the upper triangular factor U for the given square matrix A. If A is of order n, and we are doing Dolittle, i.e., making L unit lower triangular, then the natural loop is

```
for i=1:(n-1),
  compute L(:,i)
  compute U(i+1,:)
end
```

since neither the last column of L nor the first row of U needs to be calculated. More explicitly, doing the calculation in place, and with the aid of empty matrices of various sizes, the loop in an ideal MATLAB would be

```
r=1:0;
for i=1:(n-1)
   s=(i+1):n; a(s,i)    =  (a(s,i) - a(s,r)*a(r,i))/a(i,i);
   r=  1:i  ; a(i+1,s)  =  a(i+1,s) - a(i+1,r)*a(r,s);
end
```

In particular, note that, on the first pass through the loop, `r` is empty, hence `a(s,r)` is a matrix with $n-1$ (= `length(s)`) rows and no columns, but its right factor matches this, since it has no rows, hence the product is well-defined: it is a zero matrix with $n-1$ rows and 1 column, i.e., the zero column matrix of length $n-1$.

As Warren Ferguson points out, in a MATLAB without all these empty matrices, one would have to do the first pass through the loop separately. One could, of course, do it the standard way, by having the ith pass through the loop compute $L(:,i)$ and $U(i,:)$. But then both the first and the last pass through the loop would have to be done separately.

Acknowledgement. It is a pleasure to acknowledge illuminating discussions with Hans Schneider on this empty topic, and to thank Cleve Moler for the APL reference.

References

[B] N. Bourbaki, *Algèbre linéaire*, (this is chap.II of *Algèbre*, i.e., vol.IV of *Éléments de mathématique*), Hermann, Paris, 1947.

[BJ] J.A. Brown and M.A. Jenkins, The APL Identity Crisis, *APL Quote Quad* **12** (1981), 62–66.

[M] MathWorks, MATLAB User's Guide, MathWorks Inc., South Natick MA, February 1989