Regional Research Institute
West Virginia University
Technical Document Series

An Aggregation Matrix MATLAB Function

Caleb Stair

RRI TechDoc 2013-04

Date Submitted: 12/17/2013

Key Words/Codes: Aggregate/Aggregation Matrix/Input-Output

Creating an Aggregation Matrix in MatLab

Abstract. This Technical Document describes the foundations for an aggregation matrix function implemented in MATLAB, including the format and structure of the required aggregation vector used as an argument to the function. The function is passed with the N-dimensional aggregation vector as an argument. The aggregation vector contains N values ranging from 1 to M, each of which is the aggregate index corresponding to the N pre-aggregation indices. The function returns an aggregation matrix with M rows and N columns. Pre-multiplying an existing matrix with N rows by the aggregation matrix reduces the row dimensionality from N to M by adding the sectors to be aggregated. Post-multiplication by the transpose of the aggregation matrix reduces the column dimensionality from N to M accordingly.

Problem Context

Define an aggregation matrix as a k x n matrix of ones and zeroes, where k is the row dimension of the aggregated version of some un-aggregated matrix Z, which has row dimension n. The locations of ones in row i of the aggregation matrix indicate which sectors of Z will be grouped together in the aggregate sector i in the target aggregated matrix (Miller et al., 2009)

<u>*Example # 1*</u>: (n=4 and k=3) Combine additively sectors 2 and 3 of a disaggregated matrix. The aggregation matrix S that accomplishes this is

$$S = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Let Y and Y^* be the un-aggregated and aggregated vectors of initial values. The goal is to aggregate sectors 2 and 3 of the un-aggregated model. To find Y^* pre-multiply Y by the aggregation matrix S.

$$Y^* = SY = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \end{bmatrix} = \begin{bmatrix} Y_1 \\ Y_2 + Y_3 \\ Y_4 \end{bmatrix}$$

MATLAB Usage:

```
%Function aggF(r) as defined below used r as an argument and returns
aggregation matrix S by default.

Y=magic(4)
 Y =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1

r=[1 2 2 3]
 r =
     1     2     2     3

q=aggF(r)*Y   %Q is now 3 x 4
 q =
    16     2     3    13
    14    18    16    20
     4    14    15     1

q=q*aggF(r)'  %Q is now 3 x 3
 q =
    16     5    13
    14    34    20
     4    29     1
```

Example # 2: Given an aggregation scheme r for an un-aggregated matrix Z, produce the new aggregated matrix C.

$$r = \begin{bmatrix} 1 \\ 1 \\ 4 \\ 3 \\ 3 \\ 2 \end{bmatrix} \qquad Z = \begin{bmatrix} 1 & 7 & 11 & 2 & 14 & 17 \\ 8 & 6 & 19 & 2 & 3 & 16 \\ 4 & 9 & 10 & 1 & 2 & 6 \\ 19 & 18 & 22 & 3 & 5 & 9 \\ 11 & 2 & 6 & 15 & 7 & 10 \\ 12 & 8 & 3 & 5 & 13 & 1 \end{bmatrix}$$

The elements in r correspond to the destinations in the aggregated matrix of the components of the disaggregated matrix. Duplicate numbers indicate that multiple rows will be combined. (In this example, row 1 and 2 of matrix Z will be combined to form the new row 1 in matrix C. Row 3 of Z will be left as is it and will become row 4 in C). The corresponding aggregation matrix S for the aggregation scheme r is:

$$S = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Pre-multiplying Z by S will give the new aggregated matrix C.

$$C = SZ = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 7 & 11 & 2 & 14 & 17 \\ 8 & 6 & 19 & 2 & 3 & 16 \\ 4 & 9 & 10 & 1 & 2 & 6 \\ 19 & 18 & 22 & 3 & 5 & 9 \\ 11 & 2 & 6 & 15 & 7 & 10 \\ 12 & 8 & 3 & 5 & 13 & 1 \end{bmatrix} = \begin{bmatrix} 9 & 13 & 30 & 4 & 17 & 33 \\ 12 & 8 & 3 & 5 & 13 & 1 \\ 30 & 20 & 28 & 18 & 12 & 19 \\ 4 & 9 & 10 & 1 & 2 & 6 \end{bmatrix}$$

The new matrix C can be further aggregated to a 4x4 matrix by post-multiplying C by the transpose of S.

$$B = CS' = \begin{bmatrix} 9 & 13 & 30 & 4 & 17 & 33 \\ 12 & 8 & 3 & 5 & 13 & 1 \\ 30 & 20 & 28 & 18 & 12 & 19 \\ 4 & 9 & 10 & 1 & 2 & 6 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 22 & 33 & 21 & 30 \\ 20 & 1 & 18 & 3 \\ 50 & 19 & 30 & 28 \\ 13 & 6 & 3 & 10 \end{bmatrix}$$

Example # 2 Matlab equivalent:

```
r=[1 1 4 3 3 2];
S=aggF(r) %produces the aggregation matrix S
 S =
   (1,1)        1
   (1,2)        1
   (4,3)        1
   (3,4)        1
   (3,5)        1
   (2,6)        1
 %S is stored as a sparse matrix
S*Z
 ans =
     9    13    30     4    17    33
    12     8     3     5    13     1
    30    20    28    18    12    19
     4     9    10     1     2     6
%this is answer C in Example # 2
```

```
ans*S'
 ans =
    22    33    21    30
    20     1    18     3
    50    19    30    28
    13     6     3    10
 %this is answer B in Example # 2
```

Supporting Algorithm(s)/Code.

```
 function S = aggF(r)
% This function is used to produce an aggregation matrix S according to
% the row aggregation scheme r
% r is an aggregation vector like r = [1 1 2 2 3 3 ... n n]
%
% Input Variables:
%   r: the row aggregation scheme.
%
% Output Variables:
%   S: the aggregation matrix given by the aggregation scheme r.
%   (S is stored as a sparse matrix)

lr=length(r);
S=sparse(max(r),lr);
for i=1:lr
   S(r(i),i)=1;
end
```

References

Miller, R. E., & Blair, P. D. (2009). Input-output analysis: foundations and extensions. Cambridge University Press. Pg. 161-164