```
function Ac = select_cardinality(A, cardinality, comparison_method)
%SELECT_CARDINALITY filters out hyperedges based on cardinality. This
%function assumes the incidence matrix is binary.
% TODO: error checking
%
% A: (m,n) double incidence matrix
% cardinality: double, (n,1) double.
% compfun: string {'eq', 'leq', 'geq', 'in', 'out'}

% card -> double
%   1. 'eq' (default): returns hyperedges with exactly card nodes
%   2. 'leq': returns hyperedges with less than or equal to card nodes
%   3. 'geq': returns hyperedges with greater than or equal to card nodes
%
% card -> (1,n) double
%   1. 'eq' (default): for each cardinality in card, returns hyperedges
%   with exactly that many nodes. Ex: if cardinality = [2, 3, 6], then this
%   function returns all hyperedges with 2, 3, or 6 nodes.
%
% card -> (1,2) double
%   1. == (default)
%   2. range between the two numbers
%   3. range from 0 to the first number intersect range from the second
%   number to the maximum cardinality, inclusive.

arguments
    A (:,:)
    cardinality (1,:)
    comparison_method string = 'eq'
end
    edge_sizes = sum(A,1);
    if length(cardinality) == 1
        switch comparison_method
            case 'eq'
                Ac = A(:, edge_sizes == cardinality);
            case 'leq'
                Ac = A(:, edge_sizes <= cardinality);
            case 'geq'
                Ac = A(:, edge_sizes >= cardinality);
            otherwise
                % error, invalid comparison method for single size
                return
        end
    elseif length(cardinality) == 2
        c1 = min(cardinality);
        c2 = max(cardinality);

        switch comparison_method
            case 'eq'
                idx = any([edge_sizes == c1; edge_sizes == c2], 1);
                Ac = A(:, idx);
            case 'in'
```

```
                idx = all([edge_sizes >= c1; edge_sizes <= c2], 1);
                Ac = A(:, idx);
            case 'out'
                idx = any([edge_sizes <= c1; edge_sizes >= c2], 1);
                Ac = A(:, idx);
            otherwise
                % error, invalid comparison method for two sizes
                return
        end
    else
        % TODO: check that cardinality is one-dimensional array
        V = ismember(edge_sizes, cardinality);
        idx = any(V, 1);
        Ac = A(:, idx);
    end
end
```

Published with MATLAB® R2021b