

Voting system for multi-model comparison

Description, examples, and tests of vote-processing rules

William J M Probert, Sam Nicol, Matthew J Ferrari, Shou-Li Li, Katriona Shea, Michael J Tildesley, Michael C Runge

Overview

Vote-processing rules applied to decision-making in infectious disease response.

- Interventions are the “candidates”
- Processed rankings of interventions from model output are the “votes”
 - Processing depends on type of model (stochastic/deterministic), whether to remove ties, etc

Four vote processing rules outlined:

1. First-past-the-post (FPP)
2. Borda Count
3. Coombs Method
4. Alternative Vote

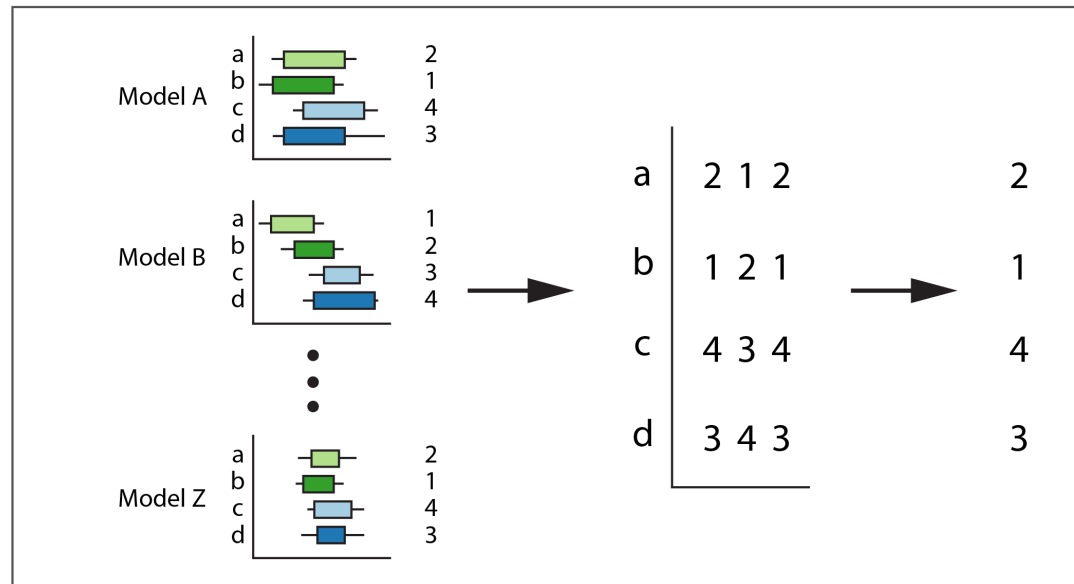


Fig 1: Schematic of a hypothetical application of the approach in this work. Model results from models A to Z provide projections from four interventions (a, b, c, d) on a particular objective (such as a measure of outbreak severity). Within each model, projections of interventions are ranked and these ranks are then processed using vote-processing rules to determine a best intervention (i.e. a ‘winner’).

Overview

- This document outlines definitions of the four vote-processing rules.
- This document outlines a selection of the tests applied to the vote-processing rules.
- The full suite of tests includes 71 tests.
 - See `voting_systems/tests/test_voting_rules.py` for all the tests.

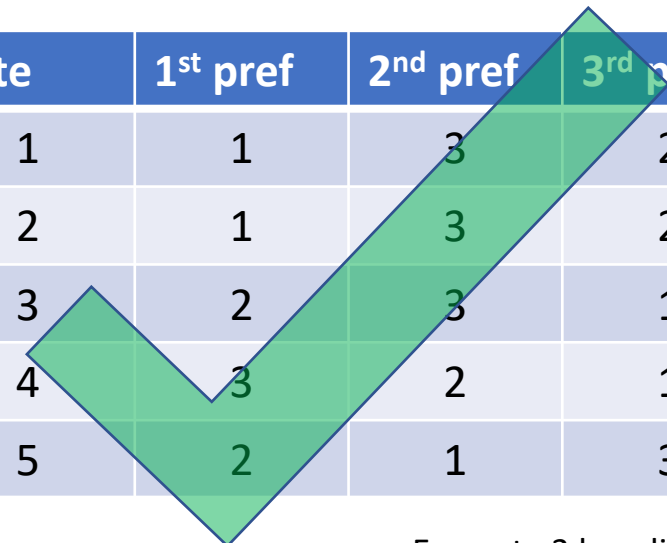
Note

Throughout the examples and code the following convention is adopted

Sets of votes are displayed as:

- rows as votes
- columns as **preference/rank**
- elements as **candidates/interventions**

Vote	1 st pref	2 nd pref	3 rd pref
1	1	3	2
2	1	3	2
3	2	3	1
4	3	2	1
5	2	1	3



E.g. vote 3 has different interpretations:

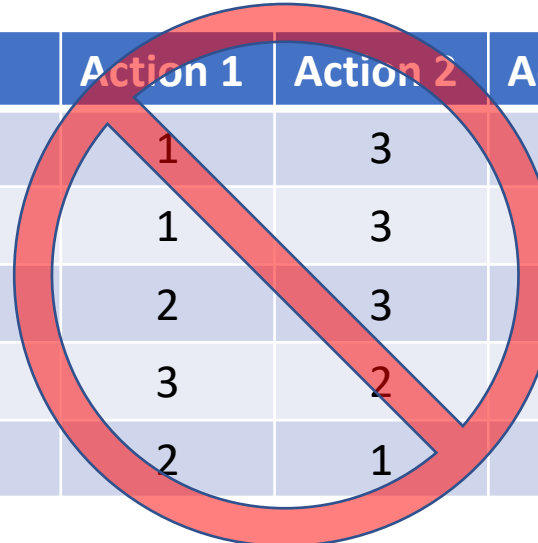
LHS table – Best is intervention 2, second is intervention 3, third is intervention 1

RHS table – Best is intervention 3, second is intervention 1, third is intervention 2

Sets of votes are **NOT** displayed as:

- rows as votes
- columns as **candidates/interventions**
- elements as **preference/rank**

Vote	Action 1	Action 2	Action 3
1	1	3	2
2	1	3	2
3	2	3	1
4	3	2	1
5	2	1	3

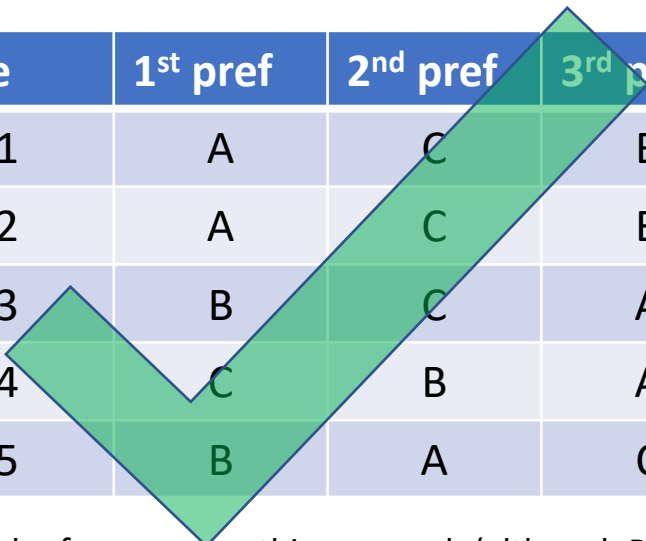


To avoid confusion, where possible, candidate/action names are used instead of indices

Sets of votes are displayed as:

- rows as votes
- columns as **preference/rank**
- elements as **candidates/interventions**

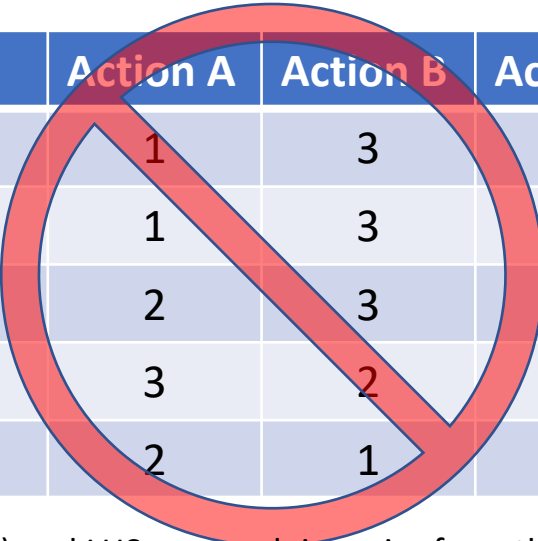
Vote	1 st pref	2 nd pref	3 rd pref
1	A	C	B
2	A	C	B
3	B	C	A
4	C	B	A
5	B	A	C



Sets of votes are **NOT** displayed as:

- rows as votes
- columns as **candidates/interventions**
- elements as **preference/rank**

Vote	Action A	Action B	Action C
1	1	3	2
2	1	3	2
3	2	3	1
4	3	2	1
5	2	1	3



Why? Several references use this approach (although Burgman et al., 2014 uses the RHS approach) and LHS approach is easier from the POV of computation (i.e. a tally of unique rows in this matrix preserves the same information as the whole ballot of votes), whereas RHS requires additional processing. Both approaches are correct but it's easier for interpretation if the method of presenting votes is consistent throughout a body of work.

1. First-past-the-post (FPP)

Description: *Voters choose a single best action. The action with the most votes wins. Absolute majority is not needed to win (just the largest proportion of first preferences).*

Examples of use: Members of Parliament of the House of Commons, UK, passage of a measure in the US Senate

Cons: votes may be seen as “wasted” (Reeve & Ware, ‘92; UK electoral reform, ‘18), criticised for resulting in majoritarian governments where only two large parties dominate elections and coalitions are rare so will ignore input from all but the most voted-for model, ties occur often when there are a small number of candidates

1. FPP – basic example

see array `exampleC_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_fpp_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	D	A	C	E	B
2	D	C	E	A	B
3	C	B	A	E	D
4	B	A	C	E	D

1. FPP – basic example

see array `exampleC_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_fpp_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	D	A	C	E	B
2	D	C	E	A	B
3	C	B	A	E	D
4	B	A	C	E	D

All preferences beyond 1st preference are ignored

(FPP is not a preferential voting system. i.e. one that uses an order of preferences)

1. FPP – basic example

see array `exampleC_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_fpp_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	D	A	C	E	B
2	D	C	E	A	B
3	C	B	A	E	D
4	B	A	C	E	D

D has most 1st preference votes: **D is the winner**

1. FPP – extended example

Voter	1 st pref
1	A
2	A
3	A
4	C
5	B
6	B
7	C

Doesn't have to be an absolute majority (i.e. $>50\%$ of all votes)

Here, A has most votes: **A is the winner**

1. FPP – example with ties

see array `exampleE_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_fpp_ties_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	B	D	A	E	C
4	C	B	A	E	D
5	B	A	C	E	D

1. FPP – example with ties

see array `exampleE_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_fpp_ties_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	B	D	A	E	C
4	C	B	A	E	D
5	B	A	C	E	D

All preferences beyond 1st preference are ignored

(FPP is not a preferential voting system. i.e. one that uses an order of preferences)

1. FPP – example with ties

see array `exampleE_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_fpp_ties_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	B	D	A	E	C
4	C	B	A	E	D
5	B	A	C	E	D

Ties in number of votes returns two winners: A and B both win

1. FPP – homogeneity

see `test_test_fpp_homogeneity()` in `voting_systems/tests/test_voting_rules.py`

- Homogeneity property:

The same winner will be determined if any number of multiples of the same sequence of votes is provided

- Homogeneity of FPP is tested using seven different ballots. For each, the ballot is repeated from 2-50 times and the returned winner is checked to be the same as when using the original ballot of votes.

2. Borda Count (BC)

Description: *Each action is given points based upon voting preferences. If there are N different actions then each action is given N points for each first preference vote, $N-1$ points for each second preference vote, and so on. Points are then tallied for each action and the action with the highest number of votes is declared a winner.*

Examples of use: Many examples in environmental decision-making in
Burgman et al., (2014), Eurovision winner

Cons: 1) Scoring systems have been criticised in decision-making contexts (Game et al., 2013) (we're trying to avoid prioritization arithmetic, i.e. 1 is better than 2 but no comment on *how much better*), 2) the most preferred candidate can lose

2. Borda Count – basic example

see array `exampleB_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_borda_count_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	A	C	E	D	B
4	C	B	A	E	D
5	C	B	A	E	D
6	B	A	C	E	D
7	B	A	C	E	D

2. Borda Count – basic example

see array `exampleB_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_borda_count_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	5	D	C	E	B
2	5	C	E	D	B
3	5	C	E	D	B
4	C	B	3	E	D
5	C	B	3	E	D
6	B	4	C	E	D
7	B	4	C	E	D

Candidate A gets 29 points ($5 + 5 + 5 + 4 + 4 + 3 + 3 = 29$)

2. Borda Count – basic example

see array exampleB_char in voting_systems/tests/test_voting_rules.py
tested by test_borda_count_winner()

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	1
2	A	C	E	D	1
3	A	C	E	D	1
4	C	4	A	E	D
5	C	4	A	E	D
6	5	A	C	E	D
7	5	A	C	E	D

Candidate A gets 29 points

Candidate B gets 21 points

and so on ...

2. Borda Count – basic example

see array exampleB_char in voting_systems/tests/test_voting_rules.py
tested by test_borda_count_winner()

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	A	C	E	D	B
4	C	B	A	E	D
5	C	B	A	E	D
6	B	A	C	E	D
7	B	A	C	E	D

Candidate	Points
A	29
B	21
C	27
D	12
E	16

$$= (5*3) + (4*2) + (3*2) + (2*0) + (1*0)$$

2. Borda Count –example with ties

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_borda_count_ties()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	C	B	A	E	D
4	C	B	A	E	D
5	B	A	C	E	D

2. Borda Count –example with ties

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_borda_count_ties()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	C	B	A	E	D
4	C	B	A	E	D
5	B	A	C	E	D

Ties in highest number
of points returns multiple
winners: A and C both win

Candidate	Points
A	20
B	15
C	20
D	9
E	11

$$= (5*2) + (4*1) + (3*2) + (2*0) + (1*0)$$

$$= (5*2) + (4*1) + (3*2) + (2*0) + (1*0)$$

2. Borda Count –example with quirk

see array `exampleD_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_borda_count_quirk_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	B	D	E	C
2	A	B	D	E	C
3	A	B	D	E	C
4	C	B	E	D	A
5	C	B	E	D	A

2. Borda Count –example with quirk

see array `exampleD_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_borda_count_quirk_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	B	D	E	C
2	A	B	D	E	C
3	A	B	D	E	C
4	C	B	E	D	A
5	C	B	E	D	A

Candidate	Points
A	17
B	20
C	13
D	13
E	12

Most preferred candidate, A, lost to B.
(Candidate A would have won under FPP, CM, AV)

2. Borda Count – homogeneity

see `test_test_borda_count_homogeneity()` in
`voting_systems/tests/test_voting_rules.py`

- Homogeneity property:

The same winner will be determined if any number of multiples of the same sequence of votes is provided

- Homogeneity of Borda Count is tested using seven different ballots. For each, the ballot is repeated from 2-50 times and the returned winner is checked to be the same as when using the original ballot of votes.

3. Coombs Method

Description: *First preference votes are counted in each round. If an action has an absolute majority ($>50\%$) of first preferences then that action is declared the winner. If no winner is declared then the action with the highest proportion of last preference votes is eliminated (Tideman, 2006). Rounds of counting and elimination are continued until a winner is found.*

3. Coombs Method (CM)

Description: *First preference votes are counted in each round. If an action has an absolute majority (>50%) of first preferences then that action is declared the winner. If no winner is declared then the action with the highest proportion of last preference votes is eliminated (Tideman, 2006). Rounds of counting and elimination are continued until a winner is found.*

Note: this is absolute majority (i.e. $>$ and not \geq), if it's a tie for first place then a subsequent round will be triggered. Same applies for AV (described next).

3. Coombs Method – basic example

see array exampleE_char in voting_systems/tests/test_voting_rules.py
tested by test_coombs_method_winner()

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	B	D	A	E	C
4	C	B	A	E	D
5	B	A	C	E	D

3. Coombs Method – basic example

see array exampleE_char in voting_systems/tests/test_voting_rules.py
tested by test_coombs_method_winner()

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	B	D	A	E	C
4	C	B	A	E	D
5	B	A	C	E	D

Round 1

- No absolute majority of 1st preferences

3. Coombs Method – basic example

see array `exampleE_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_coombs_method_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	B	D	A	E	C
4	C	B	A	E	D
5	B	A	C	E	D

Round 1

- No absolute majority of 1st preferences
- Ties in 5th pref (between B and D)

3. Coombs Method – basic example

see array exampleE_char in voting_systems/tests/test_voting_rules.py
tested by test_coombs_method_winner()

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	B	D	A	E	C
4	C	B	A	E	D
5	B	A	C	E	D

Round 1

- No absolute majority of 1st preferences
- Ties in 5th pref
 - Look at 4th pref (more 4th pref for D than B)

3. Coombs Method – basic example

see array exampleE_char in voting_systems/tests/test_voting_rules.py
tested by test_coombs_method_winner()

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	*	C	E	B
2	A	C	E	*	B
3	B	*	A	E	C
4	C	B	A	E	*
5	B	A	C	E	*

Round 1

- No absolute majority of 1st preferences
- Ties in 5th pref
 - Look at 4th pref: remove D

3. Coombs Method – basic example

see array exampleE_char in voting_systems/tests/test_voting_rules.py
tested by test_coombs_method_winner()

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	C	E	B	
2	A	C	E	B	
3	B	A	E	C	
4	C	B	A	E	
5	B	A	C	E	

Round 2

- Still no absolute majority of 1st preferences

3. Coombs Method – basic example

see array `exampleE_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_coombs_method_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	C	E	B	
2	A	C	E	B	
3	B	A	E	C	
4	C	B	A	E	
5	B	A	C	E	

Round 2

- Still no absolute majority of 1st preferences
- Ties in 4th pref (between B and E)

3. Coombs Method – basic example

see array `exampleE_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_coombs_method_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	C	E	B	
2	A	C	E	B	
3	B	A	E	C	
4	C	B	A	E	
5	B	A	C	E	

Round 2

- Still no absolute majority of 1st preferences
- Ties in 4th pref
 - Look at 3rd pref (more 3rd pref for E than B)

3. Coombs Method – basic example

see array exampleE_char in voting_systems/tests/test_voting_rules.py
tested by test_coombs_method_winner()

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	C	*	B	
2	A	C	*	B	
3	B	A	*	C	
4	C	B	A	*	
5	B	A	C	*	

Round 2

- Still no absolute majority of 1st preferences
- Ties in 4th pref
 - Look at 3rd pref: remove E

3. Coombs Method – basic example

see array exampleE_char in voting_systems/tests/test_voting_rules.py
tested by test_coombs_method_winner()

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	C	B		
2	A	C	B		
3	B	A	C		
4	C	B	A		
5	B	A	C		

Round 3

- Still no absolute majority of 1st preferences

3. Coombs Method – basic example

see array exampleE_char in voting_systems/tests/test_voting_rules.py
tested by test_coombs_method_winner()

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	C	B		
2	A	C	B		
3	B	A	C		
4	C	B	A		
5	B	A	C		

Round 3

- Still no absolute majority of 1st preferences
- Ties in 3rd pref (between B and C)

3. Coombs Method – basic example

see array exampleE_char in voting_systems/tests/test_voting_rules.py
tested by test_coombs_method_winner()

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	C	B		
2	A	C	B		
3	B	A	C		
4	C	B	A		
5	B	A	C		

Round 3

- Still no absolute majority of 1st preferences
- Ties in 3rd pref
 - Look at 2nd pref (more 2nd pref for C than B)

3. Coombs Method – basic example

see array `exampleE_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_coombs_method_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	*	B		
2	A	*	B		
3	B	A	*		
4	*	B	A		
5	B	A	*		

Round 3

- Still no absolute majority of 1st preferences
- Ties in 3rd pref
 - Look at 2nd pref: remove C

3. Coombs Method – basic example

see array exampleE_char in voting_systems/tests/test_voting_rules.py
tested by test_coombs_method_winner()

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	B			
2	A	B			
3	B	A			
4	B	A			
5	B	A			

B now has an absolute majority of 1st preferences: **B is the winner**

- Candidates were removed in this order: D, E, C

3. Coombs Method – 1st example with ties

see array `example_deadlock_planets` in `voting_systems/tests/test_voting_rules.py`
tested by `test_coombs_ties_winner()`

Ties in all rounds

Vote	1 st pref	2 nd pref	3 rd pref
1	Mercury	Venus	Mars
2	Mercury	Mars	Venus
3	Venus	Mars	Mercury
4	Venus	Mercury	Mars
5	Mars	Mercury	Venus
6	Mars	Venus	Mercury

Two votes for each candidate at
each preference level

No candidates are removed

All candidates are winners: Mars, Mercury, Venus

3. Coombs Method – 2nd example with ties

see array `example_planets` in `voting_systems/tests/test_voting_rules.py`
tested by `test_coombs_alternative_vote_comparison_winner()`

Vote	1 st pref	2 nd pref	3 rd pref
1	Mercury	Earth	Venus
2	Mercury	Earth	Venus
3	Mercury	Venus	Earth
4	Earth	Venus	Mercury
5	Earth	Venus	Mercury
6	Earth	Venus	Mercury
7	Venus	Mercury	Earth

3. Coombs Method – 2nd example with ties

see array `example_planets` in `voting_systems/tests/test_voting_rules.py`
tested by `test_coombs_alternative_vote_comparison_winner()`

Vote	1 st pref	2 nd pref	3 rd pref
1	Mercury	Earth	Venus
2	Mercury	Earth	Venus
3	Mercury	Venus	Earth
4	Earth	Venus	Mercury
5	Earth	Venus	Mercury
6	Earth	Venus	Mercury
7	Venus	Mercury	Earth

Round 1

- No absolute majority of 1st preferences

3. Coombs Method – 2nd example with ties

see array `example_planets` in `voting_systems/tests/test_voting_rules.py`
tested by `test_coombs_alternative_vote_comparison_winner()`

Vote	1 st pref	2 nd pref	3 rd pref
1	Mercury	Earth	Venus
2	Mercury	Earth	Venus
3	Mercury	Venus	Earth
4	Earth	Venus	Mercury
5	Earth	Venus	Mercury
6	Earth	Venus	Mercury
7	Venus	Mercury	Earth

Round 1

- No absolute majority of 1st preferences
- Mercury has the most 3rd pref

3. Coombs Method – 2nd example with ties

see array `example_planets` in `voting_systems/tests/test_voting_rules.py`
tested by `test_coombs_alternative_vote_comparison_winner()`

Vote	1 st pref	2 nd pref	3 rd pref
1	*	Earth	Venus
2	*	Earth	Venus
3	*	Venus	Earth
4	Earth	Venus	*
5	Earth	Venus	*
6	Earth	Venus	*
7	Venus	*	Earth

Round 1

- No absolute majority of 1st preferences
- Mercury has the most 3rd pref: remove Mercury

3. Coombs Method – 2nd example with ties

see array `example_planets` in `voting_systems/tests/test_voting_rules.py`
tested by `test_coombs_alternative_vote_comparison_winner()`

Vote	1 st pref	2 nd pref	3 rd pref
1	Earth	Venus	
2	Earth	Venus	
3	Venus	Earth	
4	Earth	Venus	
5	Earth	Venus	
6	Earth	Venus	
7	Venus	Earth	

Earth now has an absolute majority of 1st preferences: **Earth is the winner**

- Candidates were removed in this order: Mercury

Note: this same example ballot gives a different winner under the Alternative Vote

3. Coombs Method – homogeneity

see `test_test_coombs_method_homogeneity()` in
`voting_systems/tests/test_voting_rules.py`

- Homogeneity property:

The same winner will be determined if any number of multiples of the same sequence of votes is provided

- Homogeneity of Coombs Method is tested using five different ballots. For each, the ballot is repeated from 2-50 times and the returned winner is checked to be the same as when using the original ballot of votes.

4. The Alternative Vote (AV)

Description: *All first preferences are tallied; if an absolute majority of voters (>50%) choose a single action as their first preference then counting stops and this action is chosen as the winner. If no winner is found, then the action with the smallest proportion of first preferences is eliminated.*

Examples of use: Australian House of Representatives, Irish Presidential elections, Chairs of select committees in the House of Commons, UK

Limitations: tactical voting can eliminate candidate by secondary preferences

4. Alternative vote – basic example

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	A	C	E	D	B
4	C	B	A	E	D
5	C	B	A	E	D
6	B	A	C	E	D
7	B	A	C	E	D

4. Alternative vote – basic example

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	A	C	E	D	B
4	C	B	A	E	D
5	C	B	A	E	D
6	B	A	C	E	D
7	B	A	C	E	D

Round 1

- No absolute majority of 1st preferences

4. Alternative vote – basic example

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	A	C	E	D	B
4	C	B	A	E	D
5	C	B	A	E	D
6	B	A	C	E	D
7	B	A	C	E	D

Round 1

- No absolute majority of 1st preferences
- Ties in least 1st pref (between D and E, both with zero 1st pref)

4. Alternative vote – basic example

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	E	B
2	A	C	E	D	B
3	A	C	E	D	B
4	C	B	A	E	D
5	C	B	A	E	D
6	B	A	C	E	D
7	B	A	C	E	D

Round 1

- No absolute majority of 1st preferences
- Ties in least 1st pref
 - Look at 2nd pref: E has least 2nd pref

4. Alternative vote – basic example

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	*	B
2	A	C	*	D	B
3	A	C	*	D	B
4	C	B	A	*	D
5	C	B	A	*	D
6	B	A	C	*	D
7	B	A	C	*	D

Round 1

- No absolute majority of 1st preferences
- Ties in least 1st pref
 - Look at 2nd pref: remove E

4. Alternative vote – basic example

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	D	C	B	
2	A	C	D	B	
3	A	C	D	B	
4	C	B	A	D	
5	C	B	A	D	
6	B	A	C	D	
7	B	A	C	D	

Round 2

- Still no absolute majority of 1st preferences
- D has least 1st pref

4. Alternative vote – basic example

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	*	C	B	
2	A	C	*	B	
3	A	C	*	B	
4	C	B	A	*	
5	C	B	A	*	
6	B	A	C	*	
7	B	A	C	*	

Round 2

- Still no absolute majority of 1st preferences
- D has least 1st pref: remove D

4. Alternative vote – basic example

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	C	B		
2	A	C	B		
3	A	C	B		
4	C	B	A		
5	C	B	A		
6	B	A	C		
7	B	A	C		

Round 3

- Still no absolute majority of 1st preferences
- Ties in least 1st pref (between B and C)

4. Alternative vote – basic example

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	C	B		
2	A	C	B		
3	A	C	B		
4	C	B	A		
5	C	B	A		
6	B	A	C		
7	B	A	C		

Round 3

- Still no absolute majority of 1st preferences
- Ties in least 1st pref
 - Look at 2nd pref: B has least 2nd pref

4. Alternative vote – basic example

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	C	*		
2	A	C	*		
3	A	C	*		
4	C	*	A		
5	C	*	A		
6	*	A	C		
7	*	A	C		

Round 3

- Still no absolute majority of 1st preferences
- Ties in least 1st pref
 - Look at 2nd pref: remove B

4. Alternative vote – basic example

see array `exampleA_char` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_winner()`

Voter	1 st pref	2 nd pref	3 rd pref	4 th pref	5 th pref
1	A	C			
2	A	C			
3	A	C			
4	C	A			
5	C	A			
6	A	C			
7	A	C			

A now has an absolute majority of 1st preferences: **A is the winner**

- Candidates were removed in this order: E, D, B

4. Alternative Vote – example with ties

see array `example_planets` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_coombs_comparison_winner()`

Vote	1 st pref	2 nd pref	3 rd pref
1	Mercury	Earth	Venus
2	Mercury	Earth	Venus
3	Mercury	Venus	Earth
4	Earth	Venus	Mercury
5	Earth	Venus	Mercury
6	Earth	Venus	Mercury
7	Venus	Mercury	Earth

4. Alternative Vote – example with ties

see array `example_planets` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_coombs_comparison_winner()`

Vote	1 st pref	2 nd pref	3 rd pref
1	Mercury	Earth	Venus
2	Mercury	Earth	Venus
3	Mercury	Venus	Earth
4	Earth	Venus	Mercury
5	Earth	Venus	Mercury
6	Earth	Venus	Mercury
7	Venus	Mercury	Earth

Round 1

- No absolute majority of 1st preferences

4. Alternative Vote – example with ties

see array `example_planets` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_coombs_comparison_winner()`

Vote	1 st pref	2 nd pref	3 rd pref
1	Mercury	Earth	Venus
2	Mercury	Earth	Venus
3	Mercury	Venus	Earth
4	Earth	Venus	Mercury
5	Earth	Venus	Mercury
6	Earth	Venus	Mercury
7	Venus	Mercury	Earth

Round 1

- No absolute majority of 1st preferences
- Venus has the least 1st pref

4. Alternative Vote – example with ties

see array `example_planets` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_coombs_comparison_winner()`

Vote	1 st pref	2 nd pref	3 rd pref
1	Mercury	Earth	*
2	Mercury	Earth	*
3	Mercury	*	Earth
4	Earth	*	Mercury
5	Earth	*	Mercury
6	Earth	*	Mercury
7	*	Mercury	Earth

Round 1

- No absolute majority of 1st preferences
- Mercury has the most 3rd pref: remove Venus

4. Alternative Vote – example with ties

see array `example_planets` in `voting_systems/tests/test_voting_rules.py`
tested by `test_alternative_vote_coombs_comparison_winner()`

Vote	1 st pref	2 nd pref	3 rd pref
1	Mercury	Earth	
2	Mercury	Earth	
3	Mercury	Earth	
4	Earth	Mercury	
5	Earth	Mercury	
6	Earth	Mercury	
7	Mercury	Earth	

Mercury now has an absolute majority of 1st preferences: **Mercury is the winner**

- Candidates were removed in this order: Venus

Note: this same example ballot gives a different winner under the Coombs Method

4. Alternative Vote – homogeneity

see `test_test_alternative_vote_homogeneity()` in
`voting_systems/tests/test_voting_rules.py`

- Homogeneity property:

The same winner will be determined if any number of multiples of the same sequence of votes is provided

- Homogeneity of Alternative Vote is tested using five different ballots. For each, the ballot is repeated from 2-50 times and the returned winner is checked to be the same as when using the original ballot of votes.