Package: sov (via r-universe)

June 4, 2026

Title Calculate vs-SOVs and SOVs for Assemblies with D-Dimensional Voting

Version 1.0.3

Description Calculates vote-specific and traditional Shapley-Owen power indices (vs-SOVs and SOVs) for spatial voting games in one to four dimensions. Evaluates voter influence through an a posteriori analysis of relative preferences. Supports weighted voting and various voting thresholds. Compatible with ideal point estimates from NOMINATE, Optimal Classification, and 'MCMCpack'. The method builds on Bibina and Dougherty (2025) <doi:10.2139/ssrn.6324519>.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports dplyr, openxlsx, tibble, tidyr

Suggests testthat (>= 3.0.0), withr

Config/testthat/edition 3

URL https://github.com/spatial-voting-lab/sov

BugReports https://github.com/spatial-voting-lab/sov/issues

Config/pak/sysreqs libicu-dev

Repository https://spatial-voting-lab.r-universe.dev

Date/Publication 2026-03-31 19:05:58 UTC

RemoteUrl https://github.com/spatial-voting-lab/sov

RemoteRef HEAD

RemoteSha 6283fcb2f5ee166e5ad90143d15f602d98cb9fe9

Contents

---

sov *Traditional Shapley-Owen Values*

Description

This function calculates traditional Shapley-Owen values (SOVs) from package estimated inputs.

Usage

```
sov(
  estimates = NULL,
  av = NULL,
  weight_nom = FALSE,
  absolute = FALSE,
  vw = NULL,
  q = NULL,
  pr = 0.5001,
  nPoints1 = 360,
  nPoints2 = 360,
  dec = 3,
  out_dir = NULL,
  print_results = FALSE
)
```

Arguments

estimates	Estimation results from oc, wnom, or MCMCpack.
av	Attendance vector coded attend=1, "not attend"=NA.
weight_nom	If TRUE use weights estimated from wnom output, if FALSE use vector of 1's. ... For wnom only.
absolute	If TRUE, q is a scalar of yeas needed to pass proposal. If FALSE, pr is the proportion of yeas need to pass a proposal among voters attending.
vw	Vector of weights for each voter (default 1 for each member, applied later).
q	The quota of yeas need to pass a proposal under absolute k-majority rule (scalar, default absolute.maj(vw) applied later).
pr	The proportion of yea votes needed to pass a proposal among voters attending for simple k-majority rule (scalar, default: 0.5001).

nPoints1	The number of points sampled along the azimuth (the first spherical coordinate the azimuth). Sampled from 0 to 2*pi.
nPoints2	The number of points sampled along the inclination (the second and third spherical coordinate). Sampled from 0 to 2*pi.
dec	The number of decimal places reported for vs-sovs.
out_dir	The path to the output directory. This is used only when `print_results = TRUE` and must be supplied explicitly by the user.
print_results	If TRUE, print results to an excel file in out_dir; if FALSE, don't print results. In both cases, results are returned.

Value

A list with data frames containing ideal points, number of pivots, name of pivots for each direction sampled, and SOVs for each voter.

Examples

```
## --- Ideals: 5 voters in 2D -----------------------------------------------
i1 <- c( 0.7,  0.7)
i2 <- c(-0.5,  0.5)
i3 <- c(-0.7, -0.7)
i4 <- c( 0.5, -0.5)
i5 <- c( 0.0,  0.0)
ideals <- rbind(i1, i2, i3, i4, i5)
rownames(ideals) <- paste0("i", 1:5)
colnames(ideals) <- c("coord1D","coord2D")

# Build a minimal WNOM-like 'estimates' object for sov() identification
spreads <- rbind(c( 1,  0), c( 0,  1), c(-1,  0))
midpoints <- rbind(c( 0.10,  0.00), c( 0.00, -0.10), c( 0.05,  0.00))
rownames(spreads)  <- rownames(midpoints) <- paste0("RC", 1:3)
colnames(spreads)  <- colnames(midpoints) <- c("dim1","dim2")

leg <- data.frame(
 coord1D   = ideals[, 1],
 coord2D   = ideals[, 2],
 GMP       = 0.5,
 CC        = 0.5,
 row.names = rownames(ideals),
 check.names = FALSE
)
rc <- data.frame(
 GMP = rep(0.5, nrow(midpoints)),
 midpoint1D = midpoints[, 1],
 midpoint2D = midpoints[, 2],
 spread1D   = spreads[, 1],
 spread2D   = spreads[, 2],
 row.names  = rownames(midpoints),
 check.names = FALSE
)
weights <- c(1, 1)
```

```
estimates <- list(legislators = leg, rollcalls = rc, weights = weights)
class(estimates) <- "nomObject"

# Attendance: exclude i5
av <- c(1, 1, 1, 1, NA); names(av) <- rownames(ideals)
vw <- rep(1, nrow(ideals))

out_sov <- sov(
 estimates     = estimates,
 av            = av,
 absolute      = FALSE,
 pr            = 0.5001,
 vw            = vw,
 nPoints1      = 72,
 nPoints2      = 72,
 dec           = 3,
 print_results = FALSE
)

out_sov$pivot_summary
out_sov$pivot_by_angle

### Plotting (2D): label with SOVs (no normals needed here) ###
if (interactive()) {
 sov_labels2d <- setNames(out_sov$pivot_summary$sov, out_sov$pivot_summary$name)
 sov::plot_sov_geometry(ideals, label_values = sov_labels2d, digits = 3)
}
```

---

sov_user *Traditional Shapley-Owen Values with User Inputs*

Description

This function calculates traditional Shapley-Owen values (SOVs) from data provided by the user.

Usage

```
sov_user(
  ideals = NULL,
  av = NULL,
  absolute = FALSE,
  vw = NULL,
  q = NULL,
  pr = 0.5001,
  nPoints1 = 360,
  nPoints2 = 360,
  dec = 3,
  out_dir = NULL,
  print_results = FALSE
)
```

Arguments

ideals	Matrix of ideal points (legislators x dimensions)
av	Attendance vector coded attend=1, "not attend"=NA.
absolute	If TRUE, q is a scalar of yeas needed to pass proposal. If FALSE, pr is the proportion of yeas need to pass a proposal among voters attending.
vw	Vector of weights for each voter (default 1 for each member, applied later).
q	The quota of yeas need to pass a proposal under absolute k-majority rule (scalar, default absolute.maj(vw) applied later).
pr	The proportion of yea votes needed to pass a proposal among voters attending for simple k-majority rule (scalar, default: 0.5001).
nPoints1	the number of points sampled along the azimuth (the first spherical coordinate the azimuth). Sampled from 0 to 2*pi.
nPoints2	the number of points sampled along the inclination (the second and third spherical coordinate). Sampled from 0 to 2*pi.
dec	The number of decimal places reported for vs-sovs.
out_dir	The path to the output directory. This is used only when `print_results = TRUE` and must be supplied explicitly by the user.
print_results	If TRUE, print results to an excel file in out_dir; if FALSE, don't print results. In both cases, results are returned.

Value

A list with data frames containing ideal points, number of pivots, name of pivots for each direction sampled, and SOVs for each voter.

Examples

```
# Traditional SOVs in 2D
# Note: You supply only ideal points and an attendance vector (i.e., who is
# always present), not roll-call votes.

# Ideals: 5 voters in 2D
i1 <- c( 0.7,  0.7)
i2 <- c(-0.5,  0.5)
i3 <- c(-0.7, -0.7)
i4 <- c( 0.5, -0.5)
i5 <- c( 0.0,  0.0)
ideals <- rbind(i1, i2, i3, i4, i5)
rownames(ideals) <- paste0("i", 1:5)
colnames(ideals) <- c("coord1D","coord2D")

# Attendance: 1 = included, NA = excluded
# Example: exclude i5 from the analysis
av <- c(1, 1, 1, 1, NA)

## --- Equal voting weights -------------------------------------------------
vw <- rep(1, nrow(ideals))
```

```
## --- Traditional SOV (simple majority among the four voters included) ------
out_sov <- sov_user(
  ideals   = ideals,
  av       = av,
  absolute = FALSE,   # simple k-majority among included voters
  pr       = 0.5001,
  vw       = vw,
  nPoints1 = 72,      # 360 degrees divided into 72 equal sized increments
  nPoints2 = 72,
  dec      = 3
)

# aggregate results by member
out_sov$pivot_summary
# Directions examined (normals) and pivot names by direction:
out_sov$pivot_by_angle

### Plotting (2D): label with SOVs (no normals needed) ###
 if (interactive()) {
 sov_labels2d <- setNames(out_sov$pivot_summary$sov, out_sov$pivot_summary$name)
 sov::plot_sov_geometry(ideals, label_values = sov_labels2d, digits = 3)
}
```

---

vs_sov *Vote-Specific Shapley-Owen Values*

Description

This function calculates vote-specific Shapley-Owen values (vs-SOVs) from package estimated inputs.

Usage

```
vs_sov(
  estimates = NULL,
  weight_nom = FALSE,
  absolute = FALSE,
  vw = NULL,
  q = NULL,
  pr = 0.5001,
  votes = NULL,
  dec = 3,
  out_dir = NULL,
  print_results = FALSE
)
```

Arguments

estimates	Estimation results from oc, wnom, or mcmcPack.
weight_nom	If TRUE use weights estimated from wnom output, if FALSE use vector of 1's. ... For wnom only.
absolute	If TRUE, q is a scalar of yeas needed to pass proposal. If FALSE, pr is the proportion of yeas need to pass a proposal among voters attending.
vw	Vector of weights for each voter (default 1 for each member, applied later).
q	The quota of yeas need to pass a proposal under absolute k-majority rule (scalar, default absolute.maj(vw) applied later).
pr	The proportion of yea votes needed to pass a proposal among voters attending for simple k-majority rule (scalar, default: 0.5001).
votes	Vote matrix (legislators x rollcalls) coded yea=1, nay=0, and 9="attend but abstain". All other values should be NA.
dec	The number of decimal places reported for vs-sovs.
out_dir	The path to the output directory. This is used only when `print_results = TRUE` and must be supplied explicitly by the user.
print_results	If TRUE, print results to an excel file in out_dir; if FALSE, don't print results. In both cases, results are returned.

Value

A list with data frames containing ideal points, vs-SOVs for each voter, number of pivots, name of pivot(s) for each roll call, and normal vectors and angles for each roll call.

Examples

```
# VS-SOVs in 2D using W-NOMINATE OUTPUT
# vs_sov() needs an "estimates" object from OC, WNOMINATE, or MCMCpack.
# Below is a minimal WNOMINATE-like object built from 2D ideals
# and three roll calls.

# Ideals: 5 voters in 2D
i1 <- c( 0.7,  0.7)
i2 <- c(-0.5,  0.5)
i3 <- c(-0.7, -0.7)
i4 <- c( 0.5, -0.5)
i5 <- c( 0.0,  0.0)
ideals <- rbind(i1, i2, i3, i4, i5)
rownames(ideals) <- paste0("i", 1:5)
colnames(ideals) <- c("coord1D","coord2D")

# Fabricate a minimal WNOM-like object called estimates
# Spreads pick the normal directions; midpoints place the cutplane.
spreads <- rbind(
  c( 1,  0),  # RC1: normal along +x
  c( 0,  1),  # RC2: normal along +y
  c(-1,  0)   # RC3: normal along -x
```

```
)
midpoints <- rbind(
  c( 0.10,  0.00),  # RC1 cut near the origin on x
  c( 0.00, -0.10),  # RC2 cut slightly below origin on y
  c( 0.05,  0.00)   # RC3 cut near the origin on x
)
rownames(spreads)  <- rownames(midpoints) <- paste0("RC", 1:3)

# Legislators must include coord1D/coord2D plus GMP and CC fields.
# These help the function identify the type of estimate.
leg <- data.frame(
  coord1D = ideals[, 1],
  coord2D = ideals[, 2],
  GMP = 0.5,
  CC  = 0.5,
  row.names = rownames(ideals),
  check.names = FALSE
)

# Roll calls must include GMP and the WNOM fields midpoint\*D and spread\*D.
rc <- data.frame(
  GMP = rep(0.5, nrow(midpoints)),
  midpoint1D = midpoints[, 1],
  midpoint2D = midpoints[, 2],
  spread1D   = spreads[, 1],
  spread2D   = spreads[, 2],
  row.names  = rownames(midpoints),
  check.names = FALSE
)

# Dimensional weights (first must be 1); here both dimensions are equal.
weights <- c(1, 1)

# Minimal WNOM-like object
estimates <- list(
  legislators = leg,
  rollcalls   = rc,
  weights     = weights
)
class(estimates) <- "nomObject"  # not required, but reasonable

# Votes: 1=yea, 0=nay, 9=attend-no-vote, NA=absent
votes <- cbind(
  RC1 = c(1, 0, 0, 1, 9),  # include one '9' to illustrate attendance w/o voting
  RC2 = c(1, 1, 0, 0, 0),
  RC3 = c(0, 1, 1, 0, 0)
)
rownames(votes) <- rownames(ideals)

# Equal voting weights
vw <- rep(1, nrow(ideals))

# VS-SOV from WNOM-like estimates (simple majority among attendees)
```

```
out_vs <- vs_sov(
  estimates = estimates,
  votes     = votes,
  absolute  = FALSE,  # simple k-majority
  pr        = 0.5001,
  vw        = vw,
  dec       = 3
)

# aggregate results by member
out_vs$pivot_summary
# Pivot names by roll call:
out_vs$pivot_by_rc
# Normals and angles (derived from spreads & midpoints):
out_vs$nv_and_angles

### Plotting (2D): one figure with ALL normals derived from 'spreads' ###
if (interactive()) {
 vs_labels2d <- setNames(out_vs$pivot_summary$vs_sov, out_vs$pivot_summary$name)
 sov::plot_sov_geometry(ideals, normals = normals, label_values = vs_labels2d, digits = 3)
}
```

---

vs_sov_user *Vote-Specific Shapley-Owen Values with User Inputs*

Description

This function calculates vote-specific Shapley-Owen Values (vs-SOVs) from data provided by the user.

Usage

```
vs_sov_user(
  ideals = NULL,
  normals = NULL,
  midpoints = NULL,
  absolute = FALSE,
  vw = NULL,
  q = NULL,
  pr = 0.5001,
  votes = NULL,
  dec = 3,
  out_dir = NULL,
  print_results = FALSE
)
```

Arguments

ideals	Matrix of ideal points (legislators x dimensions)
normals	Matrix of normal vectors (rollcalls x dimensions)
midpoints	Matrix of Midpoints (rollcalls x dimensions), i.e. intersection of the cutplane and the unobserved normal vector
absolute	If TRUE, q is a scalar of yeas needed to pass proposal. If FALSE, pr is the proportion of yeas need to pass a proposal among voters attending.
vw	Vector of weights for each voter (default 1 for each member, applied later).
q	The quota of yeas need to pass a proposal under absolute k-majority rule (scalar, default absolute.maj(vw) applied later).
pr	The proportion of yea votes needed to pass a proposal among voters attending for simple k-majority rule (scalar, default: 0.5001).
votes	Vote matrix (legislators x rollcalls) coded yea=1, nay=0, and 9="attend but abstain". All other values should be NA.
dec	The number of decimal places reported for vs-sovs.
out_dir	The path to the output directory. This is used only when `print_results = TRUE` and must be supplied explicitly by the user.
print_results	If TRUE, print results to an excel file in out_dir; if FALSE, don't print results. In both cases, results are returned.

Value

A list with data frames containing ideal points, vs-SOVs for each voter, number of pivots, name of pivot(s) for each roll call, and normal vectors and angles for each roll call.

Examples

```
###############################################################################
##### ##### 1D Example.  Start with Inputs ##### #####
## --- Ideals: 3 voters in 1D ------------------------------------------------
i1 <-  0.7
i2 <-  0.0
i3 <- -0.7
ideals <- cbind(coord1D = c(i1, i2, i3))
rownames(ideals) <- paste0("i", 1:3)

## --- Normals: 2 roll calls (x+, x-) ----------------------------------------
nv1 <-  1   # points to the right
nv2 <- -1   # points to the left
normals <- cbind(dim1 = c(nv1, nv2))
rownames(normals) <- paste0("RC", 1:2)

## --- Votes: 1=yea, 0=nay, 9=attend-no-vote, NA=absent ---------------------
votes <- cbind(
  RC1 = c(1, 0, 0),   # i1 yea, i2 nay, i3 nay
  RC2 = c(0, 1, 1)    # i1 nay, i2 yea, i3 yea
)
```

```
rownames(votes) <- rownames(ideals)

## --- Equal voting weights -------------------------------------------------
vw <- rep(1, nrow(ideals))

##### EX1: Vote-specific SOV (simple majority among attendees in 1D) #####
out_simple <- vs_sov_user(
  ideals   = ideals,
  normals  = normals,
  votes    = votes,
  absolute = FALSE,    # simple k-majority
  pr       = 0.5001,   # strict majority of attendees
  vw       = vw,
  dec      = 3
)

# Aggregate results
out_simple$pivot_summary
out_simple$pivot_by_rc
out_simple$nv_and_angles


################################################################################
##### ##### 2D Examples.  Start with Inputs ##### #####
## --- Ideals: 5 voters in 2D ----------------------------------------------
i1 <- c( 0.7,  0.7)
i2 <- c(-0.5,  0.5)
i3 <- c(-0.7, -0.7)
i4 <- c( 0.5, -0.5)
i5 <- c( 0.0,  0.0)
ideals <- rbind(i1, i2, i3, i4, i5)
rownames(ideals) <- paste0("i", 1:5)
colnames(ideals) <- c("coord1D","coord2D")

## --- Normals: 3 roll calls (x+, y+, x-) ----------------------------------
nv1 <- c( 1, 0)
nv2 <- c( 0, 1)
nv3 <- c(-1, 0)
normals <- rbind(nv1, nv2, nv3)
rownames(normals) <- paste0("RC", 1:3)

## --- Votes: 1=yea, 0=nay, 9=attend but did not vote -----------------------
votes <- cbind(
  RC1 = c(1,0,0,1,9),
  RC2 = c(1,1,0,0,0),
  RC3 = c(0,1,1,0,0)
)
rownames(votes) <- rownames(ideals)

## --- Equal voting weights -------------------------------------------------
vw <- rep(1, nrow(ideals))

##### EX2: Simple majority (2D) -- 50 percent + epsilon among attendees #####
```

```
out_simple <- vs_sov_user(
  ideals   = ideals,
  normals  = normals,
  votes    = votes,
  absolute = FALSE,
  pr       = 0.5001,
  vw       = vw,
  dec      = 3
)

out_simple$pivot_summary
out_simple$pivot_by_rc
out_simple$nv_and_angles

### Plotting (2D): one figure with ALL normals overlaid ###
if (interactive()) {
 vs_labels2d <- setNames(out_simple$pivot_summary$vs_sov, out_simple$pivot_summary$name)
 sov::plot_sov_geometry(ideals, normals = normals, label_values = vs_labels2d, digits = 3)
}
```

Index