2026 German Stata Conference

Shapley value calculations: Implementation and illustrations

Philippe Van Kerm

Munich – June 19, 2026

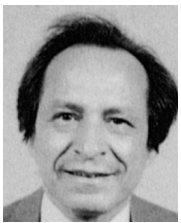
Université du Luxembourg

[Outline]

The Shapley and Owen values

Implementation with `shapowen`

Illustrations



The Shapley and Owen values

The Shapley value in a nutshell

A set of elements $N = \{1, 2, \ldots, N\}$ collectively generate some "value" (in some non-additive, non-separable way)

What is the contribution of an individual element i to the aggregate "value" collectively created by a set of elements N?

The Shapley value in a nutshell

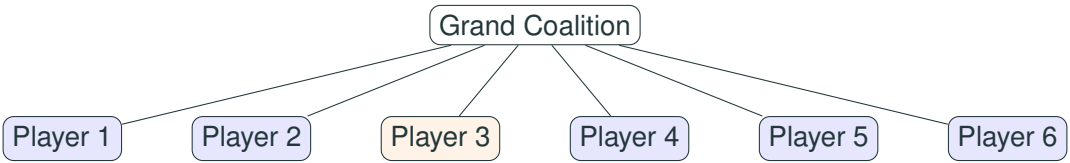
A set of elements $N = \{1, 2, \ldots, N\}$ collectively generate some "value" (in some non-additive, non-separable way)

What is the contribution of an individual element i to the aggregate "value" collectively created by a set of elements N?

The Shapley value in a nutshell

A set of elements $N = \{1, 2, \ldots, N\}$ collectively generate some "value" (in some non-additive, non-separable way)

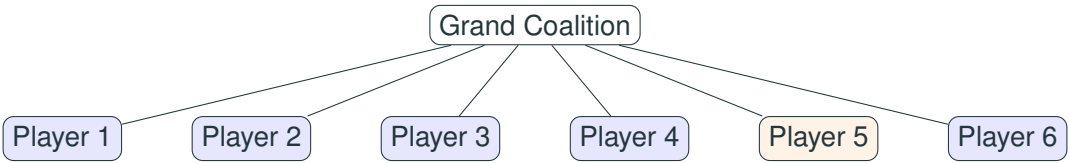
What is the contribution of an individual element i to the aggregate "value" collectively created by a set of elements N?



The Shapley value in a nutshell

A set of elements $N = \{1, 2, \ldots, N\}$ collectively generate some "value" (in some non-additive, non-separable way)

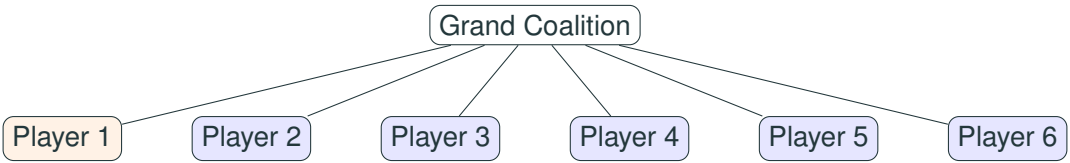
What is the contribution of an individual element i to the aggregate "value" collectively created by a set of elements N?



The Shapley value in a nutshell

A set of elements $N = \{1, 2, \ldots, N\}$ collectively generate some "value" (in some non-additive, non-separable way)

What is the contribution of an individual element i to the aggregate "value" collectively created by a set of elements N?



Many concrete examples

- Cost sharing: payment of individual users to total cost functions (cf. the “airport problem”)
- Bargaining power of political parties in coalition formation
- Contribution of individual covariates to a regression model’s R^2
- Contribution of individual predictors to a “black box” machine learning predictive or classification model (The ‘ShAP’ value)
- Contribution of different sources of income or groups of individuals to inequality in household income (many applications to income distribution analysis (Shorrocks, 2013))
- (Contributions of individual players to a football team performance? No – other tools needed.)
- ...

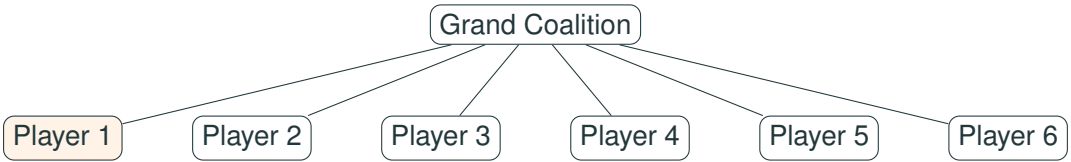
Typical but imperfect solutions

1. Sequentially add (or remove) a player and measure the marginal change in the value function

   - The contribution of a player i depends on its position in the introduction sequence

Typical but imperfect solutions

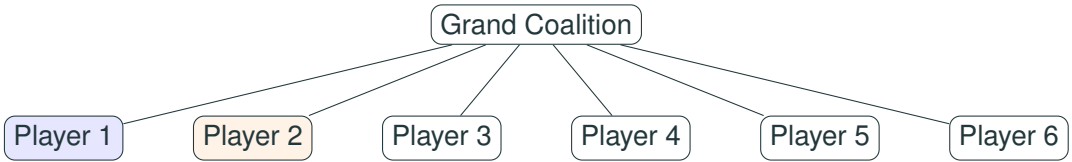
1. Sequentially add (or remove) a player and measure the marginal change in the value function



- The contribution of a player i depends on its position in the introduction sequence

Typical but imperfect solutions

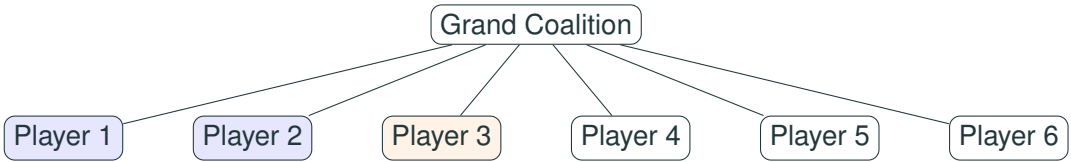
1. Sequentially add (or remove) a player and measure the marginal change in the value function



- The contribution of a player i depends on its position in the introduction sequence

Typical but imperfect solutions

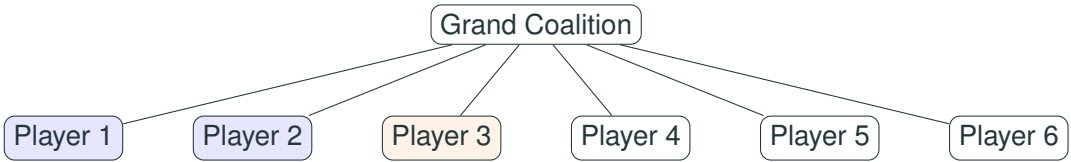
1. Sequentially add (or remove) a player and measure the marginal change in the value function



- The contribution of a player i depends on its position in the introduction sequence

Typical but imperfect solutions

1. Sequentially add (or remove) a player and measure the marginal change in the value function



- The contribution of a player i depends on its position in the introduction sequence

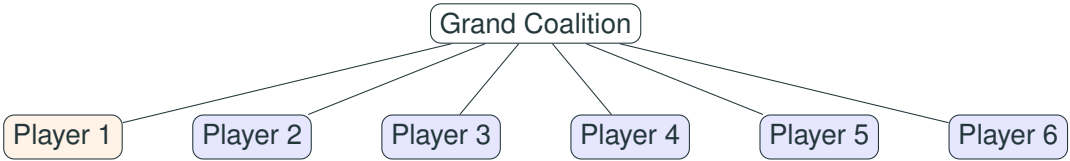
Typical but imperfect solutions

2. Remove a single player i and measure the marginal change in the value function when player i is removed from group

- The contribution of a player i depends on its position in the introduction sequence

Typical but imperfect solutions

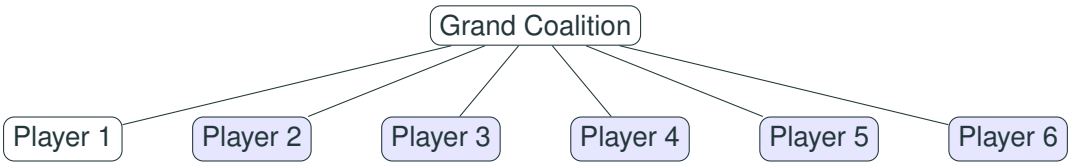
2. Remove a single player i and measure the marginal change in the value function when player i is removed from group



- The contribution of a player i depends on its position in the introduction sequence

Typical but imperfect solutions

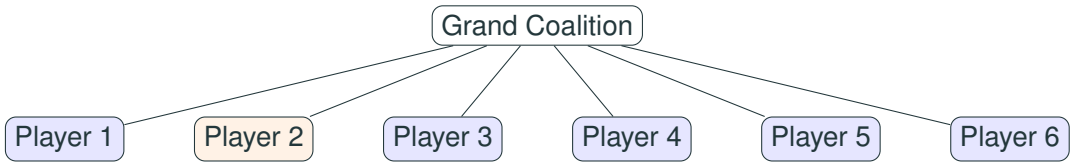
2. Remove a single player i and measure the marginal change in the value function when player i is removed from group



- The contribution of a player i depends on its position in the introduction sequence

Typical but imperfect solutions

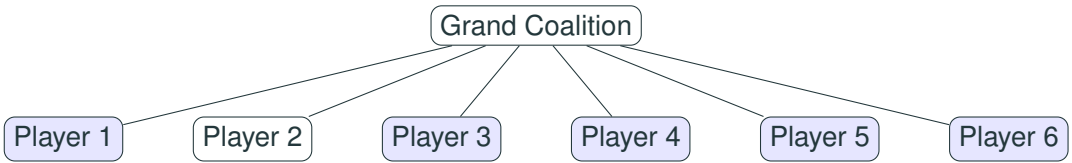
2. Remove a single player i and measure the marginal change in the value function when player i is removed from group



- The contribution of a player i depends on its position in the introduction sequence

Typical but imperfect solutions

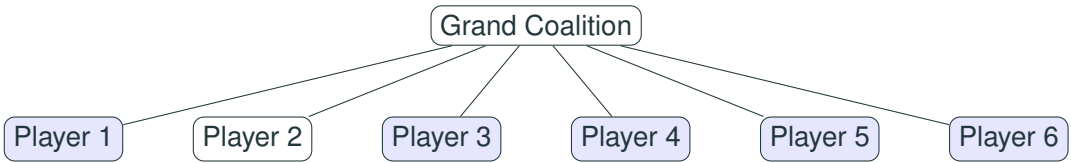
2. Remove a single player i and measure the marginal change in the value function when player i is removed from group



- The contribution of a player i depends on its position in the introduction sequence

Typical but imperfect solutions

2. Remove a single player i and measure the marginal change in the value function when player i is removed from group



- The contribution of a player i depends on its position in the introduction sequence

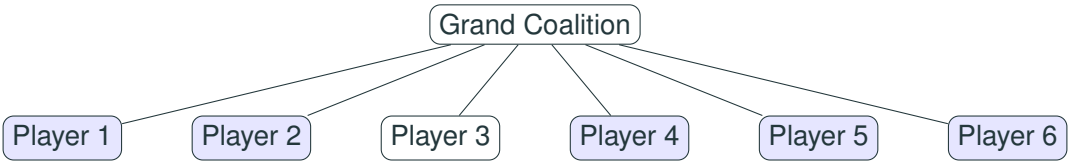
The Shapley value formula

- A set of elements ('players') $N = \{1, 2, \ldots, n\}$. N is the "grand coalition"
- A characteristic function $v : 2^N \to \mathbb{R}$ returns the collective output of any coalition formed by elements of N
- The Shapley value (Shapley, 1953) for element i is given by

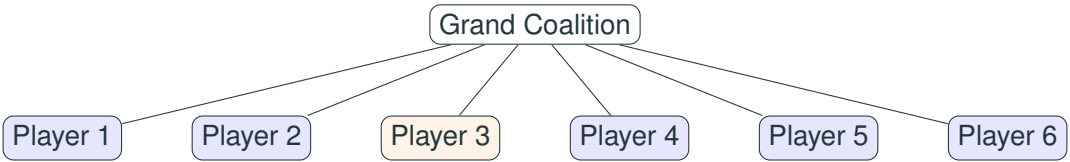
$$\phi_i = \phi(i; v, N) = \underbrace{\sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(n - |S| - 1)!}{n!}}_{\text{weighted sum across coalitions}} \underbrace{(v(S \cup \{i\}) - v(S))}_{\text{marginal contribution to coalition } S}$$

$\Longrightarrow$ equivalently: average marginal contribution of i across all *permutation* of elements in N

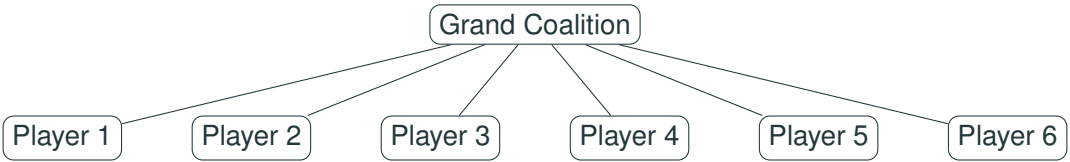
The Shapley value formula



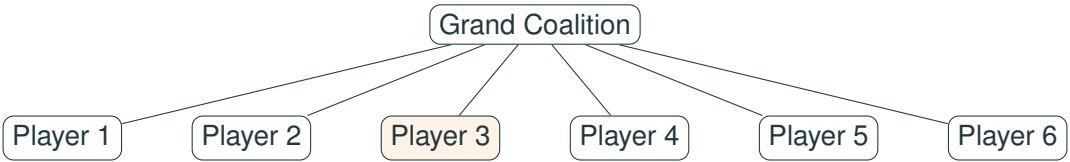
The Shapley value formula



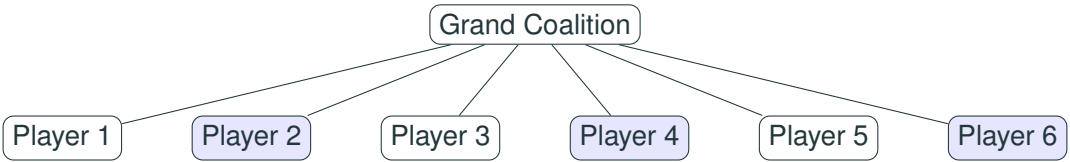
The Shapley value formula



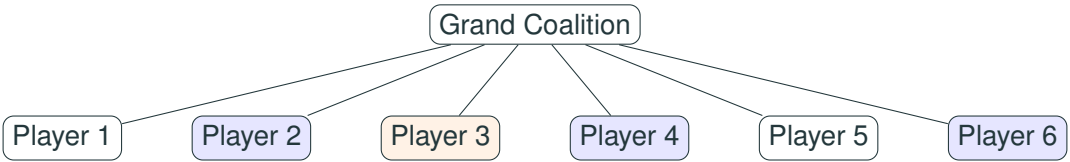
The Shapley value formula



The Shapley value formula



The Shapley value formula



Four key properties

- *Efficiency*: Additive decomposability

$$\sum_{i \in N} \phi_i = \upsilon(N)$$

- *Symmetry*: Equal marginal contributions imply equal ϕ_i
- *Dummy player*: Players contributing a fixed constant k (possibly 0) to any coalition get $\phi_i = k$
- *Additivity*: Contributions to additive games are additive themselves

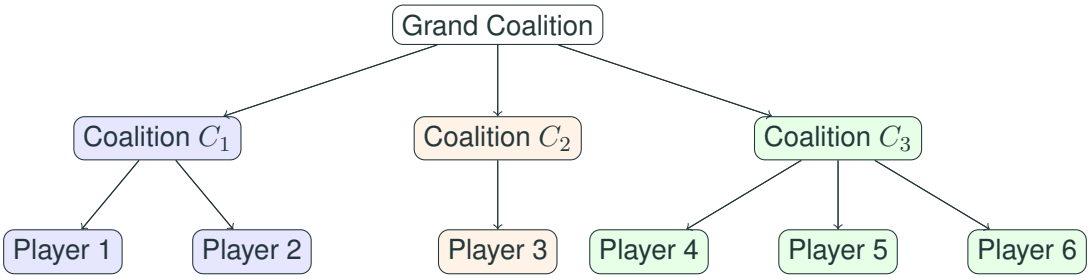
Nested structures: The Owen value (Owen, 1977)

Consider now preset (sub-)coalitions within the 'grand coalition': $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ is a partition of N into disjoint coalitions

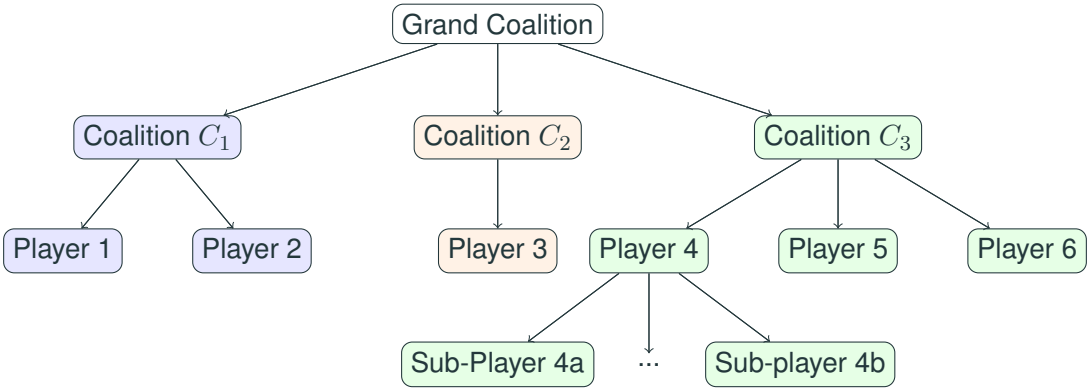
For example

- blocks of 'similar' covariates (e.g., demographic attributes, labour market characteristics and regional environment)
- blocks of 'similar' political parties (e.g., left, center, right?)
- blocks of 'similar' sources of income (public transfers vs market incomes)
- ...

Nested structures: The Owen value (Owen, 1977)



Nested structures: The Owen value (Owen, 1977)



Nested structures: The Owen value (Owen, 1977)

The Owen value is the composition of Shapley values for each preset sub-coalition

$$\psi_i(v, \mathcal{C}) = \phi_i(v_{C_j}) \cdot \phi_{C_j}(v^{\mathcal{C}})$$

- Recursive additivity: the contributions of each player in a sub-coalition add up to the contribution of the sub-coalition to the grand coalition
- Evaluate the Shapley value of a player in a sub-coalition for all possible coalitions of sub-coalitions...
- ... for any level of nested structures

Implementation with `shapowen`

[Outline]

The Shapley and Owen values

Implementation with `shapowen`

Illustrations

`shapowen` – Simplified syntax

`shapowen` evaluates Shapley and Owen values for arbitrary Stata instructions provided these

1. take input 'players' specification as (some form of) a *list*
2. return evaluation (of the value function, ϕ in `r()` or `e()` scalars or matrices (or functions thereof)

Simplified syntax diagram

`shapowen` *list-of-items*

[, scalarexpressions(*string*) matrixexpressions(*string*) substitution(*string*) ...]:

cmd ... @ ...

(Syntax borrowed from the package `shapley` available on SSC (Kolenikov, 2000).)

`shapowen` – Simplified syntax

`shapowen` evaluates Shapley and Owen values for arbitrary Stata instructions provided these

1. take input 'players' specification as (some form of) a *list*
2. return evaluation (of the value function, ϕ in `r()` or `e()` scalars or matrices (or functions thereof)

Simplified syntax diagram

`shapowen` *list-of-items*

[, scalarexpressions(*string*) matrixexpressions(*string*) substitution(*string*) ...]:

cmd ... @ ...

(Syntax borrowed from the package `shapley` available on SSC (Kolenikov, 2000).)

`shapowen` – Simplified syntax

`shapowen` evaluates Shapley and Owen values for arbitrary Stata instructions provided these

1. take input 'players' specification as (some form of) a *list*
2. return evaluation (of the value function, ϕ in `r()` or `e()` scalars or matrices (or functions thereof)

Simplified syntax diagram

`shapowen` *list-of-items*

[, scalarexpressions(*string*) matrixexpressions(*string*) substitution(*string*) ...]: *cmd ... @ ...*

(Syntax borrowed from the package `shapley` available on SSC (Kolenikov, 2000).)

`shapowen` – Simplified syntax

`shapowen` evaluates Shapley and Owen values for arbitrary Stata instructions provided these

1. take input 'players' specification as (some form of) a *list*
2. return evaluation (of the value function, ϕ in `r()` or `e()` scalars or matrices (or functions thereof)

Simplified syntax diagram

`shapowen` *list-of-items*

[, scalarexpressions(*string*) matrixexpressions(*string*) substitution(*string*) ...]: *cmd ... @ ...*

(Syntax borrowed from the package `shapley` available on SSC (Kolenikov, 2000).)

Nested list of items

Nested structures are specified in *list-of-items* by grouping items within brackets, e.g,:

```
(a b) c (d e f)
```

or

```
a (b c (d e f) (g h) i) ((j k l) m) n ''o p q''
```

NB: grouping by double-quotes forms an unbreakable item – here `o p q` are never evaluated separately.

Options

options	Description
Main	
scalarexpressions(*string*)	Scalar-valued expressions that are evaluated after each call to *cmd*
matrixexpressions(*string*)	Matrix-valued expressions that are evaluated after each call to *cmd*
Variants	
separator(*string*)	Separate items by *string* instead of the default blank space when substituting in @
substitution(*string*)	Substitute instead of remove elements excluded from a coalition in the call to *cmd*
Treatment of special cases	
emptysubstitute(*string*)	Use *string* instead of an empty string when substituting the empty set in @
emptyvalue(*string*)	Use *string* as output value (for all output expressions) when evaluating *cmd* on the empty set
errorvalue(*string*)	Use *string* as output value (for all output expressions) if execution of *cmd* fails after substitution
Approximate calculation	
approximation(# [, *opts*])	Use approximation algorithm based on # random permutations
Output and storage options	
frame	Store estimates of all combinations of items in new frame called ShapOwen
showfull	Display output of call to *cmd* with the full set of elements
showempty	Display output of call to *cmd* with the empty set
trace	Display output of all calls to *cmd*
nodots	Suppress display of progression dots
treemap	Plot results as tree map
treemapopts(*string*)	Tree map plot options

`shapowen` – Implementation comments

- Simple (non-nested) Shapley value calculations are relatively straightforward to implement – `shapowen` mainly does 'bookkeeping' on behalf of the user
- Nested structures and Owen values *are* (very) significantly more difficult to deal with
  - » `shapowen` leverages "advanced" Mata features such as classes (objects), structures, pointers and recursivity
- Speed is potentially an issue: the number of evaluations increases exponentially with the number of 'players'
  - » *cmd* needs to be fast or n needs to be relatively small

Addressing speed: approximation algorithms

- Approximation algorithms: evaluate marginal contributions $(v(S \cup \{i\}) - v(S))$ on a random subset of all potential coalitions
- `shapowen` implements a permutation-based algorithm:
  - Shapley value is *also* the average marginal contribution of i across all possible *permutations* of elements in the grand coalition
  - Monte Carlo approximation:
    1. take a random permutation of players/elements and evaluate marginal contributions in that sequence
    2. (optionally) evaluate marginal contributions on the inverse permutation (antithetic sampling)
    3. repeat for a (large) number of permutations
    4. approximate Shapley value of each element as average of their marginal contributions across permutations

Addressing speed: approximation algorithms

- Approximation algorithms: evaluate marginal contributions $(v(S \cup \{i\}) - v(S))$ on a random subset of all potential coalitions
- `shapowen` implements a permutation-based algorithm:
  - » Shapley value is *also* the average marginal contribution of i across all possible *permutations* of elements in the grand coalition
  - » Monte Carlo approximation:
    1. take a random permutation of players/elements and evaluate marginal contributions in that sequence
    2. (optionally) evaluate marginal contributions on the inverse permutation (antithetic sampling)
    3. repeat for a (large) number of permutations
    4. approximate Shapley value of each element as average of their marginal contributions across permutations

Addressing speed: approximation algorithms

- Approximation algorithms: evaluate marginal contributions $(v(S \cup \{i\}) - v(S))$ on a random subset of all potential coalitions
- `shapowen` implements a permutation-based algorithm:
  - » Shapley value is *also* the average marginal contribution of i across all possible *permutations* of elements in the grand coalition
  - » Monte Carlo approximation:
    1. take a random permutation of players/elements and evaluate marginal contributions in that sequence
    2. (optionally) evaluate marginal contributions on the inverse permutation (antithetic sampling)
    3. repeat for a (large) number of permutations
    4. approximate Shapley value of each element as average of their marginal contributions across permutations

Illustrations

Example 1: Ascribing covariate contributions to a regression's R^2

Example 1: Ascribing covariate contributions to a regression's R^2

```
. sysuse nlsw88
(NLSW, 1988 extract)

. gen lnw = ln(wage)

. regress lnw i.collgrad i.race c.age##c.age i.south
```

Source	SS	df	MS	Number of obs	=	2,246
				F(6, 2239)	=	59.04
Model	101.246931	6	16.8744885	Prob > F	=	0.0000
Residual	639.922178	2,239	.285807136	R-squared	=	0.1366
				Adj R-squared	=	0.1343
Total	741.169109	2,245	.330142142	Root MSE	=	.53461

lnw	Coefficient	Std. err.	t	P>|t|	[95% conf.	interval]
collgrad						
College grad	.4243876	.02664	15.93	0.000	.3721459	.4766293
race						
Black	-.0800319	.0268782	-2.98	0.003	-.1327407	-.0273232
Other	-.0184225	.1058507	-0.17	0.862	-.2259982	.1891532
age	.0966382	.1008467	0.96	0.338	-.1011246	.294401
c.age#c.age	-.0012791	.0012771	-1.00	0.317	-.0037835	.0012253
south						
South	-.1799692	.0237621	-7.57	0.000	-.2265672	-.1333712
_cons	.0536509	1.981686	0.03	0.978	-3.832483	3.939785

Example 1: Ascribing covariate contributions to a regression's R^2

```
. shapowen i.collgrad i.race c.age##c.age i.south , sca(e(r2_a)) :  regress lnw @
.................
Instruction:  regress lnw @
Items list:   i.collgrad i.race c.age##c.age i.south
Number of evaluations required:   16
 ---  e(r2_a) ---               Empty set value:        0                    Full set value:  .134291
```

		Nominal contribution					Relative contribution				
		Shapley -Owen	Banzhaf	First	Last	Seq.	Shapley -Owen	Banzhaf	First	Last	Seq.
[0]	FULL	.134291					1				
[1]	i.collgrad	.099846	.099654	.102766	.097694	.102766	.743508	.742079	.765248	.72748	.765248
[1]	i.race	.008584	.008411	.015207	.002653	.009181	.063923	.062636	.113237	.019755	.06837
[1]	c.age##c.age	.0003	.000322	.000371	.000137	.000561	.002231	.002401	.002762	.001021	.004176
[1]	i.south	.025561	.02537	.030101	.021783	.021783	.190339	.188919	.224151	.162206	.162206

Example 1: Ascribing covariate contributions to a regression's R^2

```
. return list

scalars:
              r(neval) =  16
                  r(K) =  4

macros:
              r(stats) : "e(r2_a) e(r2_a) e(r2_a) e(r2_a) e(r2_a) e(r2_a) e(r2_a) e(r2_a) e(r2_a) e(r2_a) e(r2_a) e(r2_a) e(r2_a) e(r2_a) e(r2_a) e(r
              r(items) : "i.collgrad i.race c.age##c.age i.south"
                r(cmd) : "regress lnw @"
              r(item5) : "i.south"
              r(item4) : "c.age##c.age"
              r(item3) : "i.race"
              r(item2) : "i.collgrad"
              r(item1) : "i.collgrad i.race c.age##c.age i.south"

matrices:
              r(Depth) :  5 x 1
       r(relSequence) :  5 x 1
          r(Sequence) :  5 x 1
           r(relLast) :  5 x 1
              r(Last) :  5 x 1
         r(relFirst) :  5 x 1
             r(First) :  5 x 1
        r(relBanzhaf) :  5 x 1
           r(Banzhaf) :  5 x 1
         r(relShap0w) :  5 x 1
            r(Shap0w) :  5 x 1
```

Example 1: Ascribing covariate contributions to a regression's R^2

(multiple measures of interest)

```
. shapowen i.collgrad i.race c.age##c.age i.south , sca(e(r2) e(r2_a)):  regress lnw @
.................
Instruction:  regress lnw @
Items list:   i.collgrad i.race c.age##c.age i.south
Number of evaluations required:   16
```

--- e(r2) --- Empty set value: 0 Full set value: .136604

		Nominal contribution					Relative contribution				
		Shapley -Owen	Banzhaf	First	Last	Seq.	Shapley -Owen	Banzhaf	First	Last	Seq.
[0]	FULL	.136604					1				
[1]	i.collgrad	.100129	.099936	.103165	.097862	.103165	.732984	.731575	.755212	.716388	.755212
[1]	i.race	.009404	.00923	.016084	.003419	.009968	.068839	.067565	.117741	.025032	.072973
[1]	c.age##c.age	.001128	.00115	.001261	.000908	.001351	.00826	.008419	.009234	.006647	.009887
[1]	i.south	.025944	.025752	.030533	.02212	.02212	.189918	.188516	.223517	.161927	.161927

--- e(r2_a) --- Empty set value: 0 Full set value: .134291

		Nominal contribution					Relative contribution				
		Shapley -Owen	Banzhaf	First	Last	Seq.	Shapley -Owen	Banzhaf	First	Last	Seq.
[0]	FULL	.134291					1				
[1]	i.collgrad	.099846	.099654	.102766	.097694	.102766	.743508	.742079	.765248	.72748	.765248
[1]	i.race	.008584	.008411	.015207	.002653	.009181	.063923	.062636	.113237	.019755	.06837
[1]	c.age##c.age	.0003	.000322	.000371	.000137	.000561	.002231	.002401	.002762	.001021	.004176
[1]	i.south	.025561	.02537	.030101	.021783	.021783	.190339	.188919	.224151	.162206	.162206

Example 1: Ascribing covariate contributions to a regression's R^2

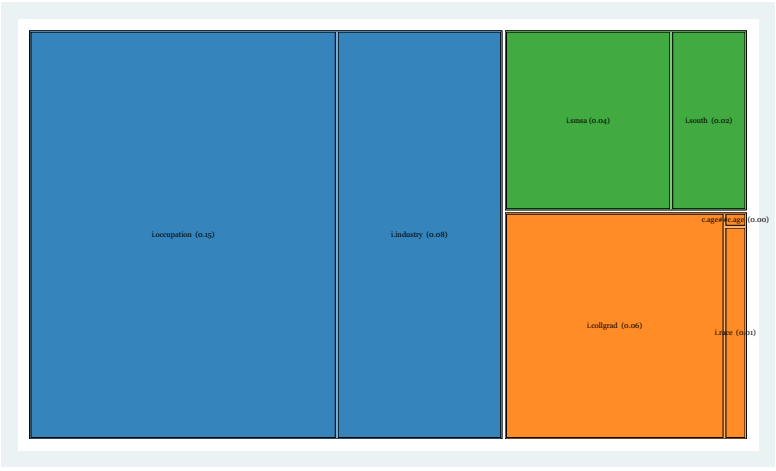
(fully interacted model – using the `sep()` option)

```
. shapowen i.collgrad c.age##c.age i.race i.south , sca(e(r2_a)) sep(##) :  regress lnw @
.................
Instruction:  regress lnw @
Items list:   i.collgrad c.age##c.age i.race i.south
Number of evaluations required:   16
 ---  e(r2_a) ---               Empty set value:        0                      Full set value:  .146942
```

		Nominal contribution					Relative contribution				
		Shapley -Owen	Banzhaf	First	Last	Seq.	Shapley -Owen	Banzhaf	First	Last	Seq.
[0]	FULL	.146942					1				
[1]	i.collgrad	.102677	.102816	.102766	.102032	.102766	.69876	.699707	.699362	.69437	.699362
[1]	c.age##c.age	.000255	.000409	.000371	-.000476	-.000441	.001735	.002782	.002524	-.003241	-.003
[1]	i.race	.014596	.014906	.015207	.012742	.016661	.099329	.101442	.103488	.086717	.113384
[1]	i.south	.029414	.029607	.030101	.027956	.027956	.200176	.201487	.204852	.190255	.190255

Note: Leveraging A. Naqvi's `treemap` for visualising nested structure

```
shapowen (i.race i.collgrad c.age##c.age) (i.industry i.occupation) (i.south i.smsa) ///
  , sca(e(r2_a)): regress lnw @
```



Example 2: Assessing mediating effects

Example 2: Assessing mediating effects

```
. shapowen i.collgrad c.age##c.age i.south , sca(_b[2.race]) :  regress lnw i.race @
........
Instruction:  regress lnw i.race @
Items list:   i.collgrad c.age##c.age i.south
Number of evaluations required:   8
 ---  _b[2.race] ---            Empty set value: -.164597                    Full set value: -.080032
```

		Nominal contribution					Relative contribution				
		Shapley-Owen	Banzhaf	First	Last	Seq.	Shapley-Owen	Banzhaf	First	Last	Seq.
[0]	FULL	.084565					1				
[1]	i.collgrad	.034787	.034778	.033757	.035856	.033757	.411368	.411254	.399181	.424009	.399181
[1]	c.age##c.age	-.002953	-.002963	-.003076	-.002792	-.002742	-.03492	-.035033	-.036371	-.033015	-.032427
[1]	i.south	.05273	.052721	.051949	.05355	.05355	.623552	.623438	.614311	.633247	.633247

Example 3: Shapley-Shubik measure of (political) power

- The Shapley-Shubik measure determines the negotiation power of political parties when they need to form a coalition after an election
- Each party is endowed with a number of seats – parties must negotiate to form a coalition holding the majority of seats
- The value function of a coalition of players is simply 0 (no majority) or 1 (majority)

Example 3: Shapley-Shubik measure of (political) power

Example from the 2023 national elections in Luxembourg

```
. list Party1 ofSeats
```

	Party1	ofSeats
1.	ADR	5
2.	CSV	21
3.	DK	0
4.	DL	0
5.	DP	14
6.	G	4
7.	KPL	0
8.	L	2
9.	LF	0
10.	LSAP	11
11.	Pirat	3
12.	Volt	0

Example 3: Shapley-Shubik measure of (political) power

Here we need a little `program` to evaluate the value function

```
. cap pr drop major

. pr def major , rclass
  1.          syntax [anything] [if] [, threshold(real 30) ]
  2.          marksample touse
  3.          loc t = 0
  4.          foreach el of local anything {
  5.                  su ofSeats if Party1==`"`el'"' & `touse' , mean
  6.                  if (r(N)>0) loc t = `t' + r(mean)
  7.          }
  8.          loc m = `t'>`threshold'
  9.          di "`m'"
 10.          return scalar m = `m'
 11. end

. major ADR DP
0

. major ADR DP CSV
1
```

Example 3: Shapley-Shubik measure of (political) power

```
. levelsof Party1 if ofSeats>0, loc(ps) clean
ADR CSV DP G L LSAP Pirat

. shapowen 'ps'  , sca(r(m)) : major @  if ofSeats>0
.....................................................................................................................
Instruction:  major @  if ofSeats>0
Items list:   ADR CSV DP G L LSAP Pirat
Number of evaluations required:   128
 ---  r(m) ---                  Empty set value:        0                      Full set value:        1
```

		Nominal contribution					Relative contribution				
		Shapley-Owen	Banzhaf	First	Last	Seq.	Shapley-Owen	Banzhaf	First	Last	Seq.
[0]	FULL	1					1				
[1]	ADR	.061905	.125	0	0	0	.061905	.125	0	0	0
[1]	CSV	.411905	.65625	0	0	0	.411905	.65625	0	0	0
[1]	DP	.211905	.34375	0	0	1	.211905	.34375	0	0	1
[1]	G	.045238	.09375	0	0	0	.045238	.09375	0	0	0
[1]	L	.028571	.0625	0	0	0	.028571	.0625	0	0	0
[1]	LSAP	.211905	.34375	0	0	0	.211905	.34375	0	0	0
[1]	Pirat	.028571	.0625	0	0	0	.028571	.0625	0	0	0

Example 4: Inequality contributions of sources of income

- Let us measure inequality of household disposable income in a country by the Gini coefficient
- Disposable income is the sum of multiple sources: e.g., market incomes (labour incomes, capital incomes), public transfers (pensions, social transfers), minus taxes and social security contributions
- Question: *What is the contribution of each of these sources to overall inequality?* (see, e.g., Sastre and Trannoy, 2002, Trannoy and Sastre, 2002, Chantreuil and Trannoy, 2013, Chantreuil et al., 2019)

Example 4: Inequality contributions of sources of income

```
. sgini  Lh Lo K  PB T O [aw=hpwgt] if period==2 , sourcedecomp   // ssc install sgini
Gini coefficient for Lh, Lo, K, PB, T, O
Note: T has 5883 negative observations (used in calculations).
```

Variable	v=2
Lh	0.5515
Lo	0.6249
K	0.9316
PB	0.5599
T	-0.5297
O	0.9646

```
Decomposition by source:
  TOTAL =  Lh +  Lo +  K +  PB +  T +  O
```

Example 4: Inequality contributions of sources of income

...
Parameter: v=2

Variable	Share s	Coeff. g	Corr. r	Conc. c=g*r	Contri. s*g*r	%Contri. s*g*r/G	Elasticity s*g*r/G-s
Lh	0.5668	0.5515	0.6910	0.3811	0.2160	0.8464	0.2796
Lo	0.4177	0.6249	0.7587	0.4741	0.1980	0.7760	0.3583
K	0.0342	0.9316	0.6583	0.6133	0.0210	0.0822	0.0480
PB	0.3273	0.5599	-0.1094	-0.0613	-0.0200	-0.0786	-0.4058
T	-0.3500	-0.5297	-0.8616	0.4564	-0.1597	-0.6258	-0.2759
O	0.0040	0.9646	-0.0130	-0.0125	-0.0000	-0.0002	-0.0042
TOTAL	1.0000	0.2552	1.0000	0.2552	0.2552	1.0000	0.0000

Example 4: Inequality contributions of sources of income

```
. shapowen Lh Lo K  PB T O , scalar(r(coeff)) error(0) : ///
>         sgini @ [aw=hpwgt] , source
..Execution of --          sgini  [aw=hpwgt] , source -- failed (error 100)
Set error value 0 used
.............................................................
Instruction:  sgini @ [aw=hpwgt] , source
Items list:   Lh Lo K  PB T O
 ---  r(coeff) ---              Empty set value:         0                  Full set value:  .2552038
```

	Nominal contribution				Relative contribution			
	Shapley -Owen	Banzhaf -Owen	First	Last	Shapley -Owen	Banzhaf -Owen	First	Last
FULL	.2552038				1			
Lh	-.7172844	-.6949358	.551542	-.3362231	-2.810634	-2.723062	2.161182	-1.317469
Lo	-.3582809	-.2313472	.6249195	-.1139094	-1.403901	-.9065192	2.448708	-.4463466
K	1.466194	2.491113	.9316415	.0072015	5.745189	9.761267	3.650578	.0282187
PB	-.1321958	.2120674	.559928	-.2501125	-.5180009	.8309725	2.194042	-.98005
T	.0146824	.6560417	-.5296846	-.0594907	.0575319	2.570658	-2.075536	-.2331106
O	-.0179117	-.3026071	.964634	-.0015981	-.070186	-1.185747	3.779857	-.0062619

Example 4: Inequality contributions of sources of income

```
. gen zero = 0.01
. shapowen Lh Lo K  PB T O , scalar(r(coeff)) : ///
>         sgini zero @ [aw=hpwgt]  , source
.....................................................................
Instruction:  sgini zero @ [aw=hpwgt]  , source
Items list:   Lh Lo K  PB T O
```

--- r(coeff) --- Empty set value: 0 Full set value: .2552037

	Nominal contribution				Relative contribution			
	Shapley -Owen	Banzhaf -Owen	First	Last	Shapley -Owen	Banzhaf -Owen	First	Last
FULL	.2552037				1			
Lh	-.7172473	-.6948817	.5515417	-.3362227	-2.810489	-2.722851	2.161182	-1.317468
Lo	-.3582452	-.2312952	.6249191	-.1139092	-1.403762	-.9063157	2.448707	-.4463463
K	1.466184	2.491093	.9316334	.0072015	5.74515	9.761195	3.650548	.0282187
PB	-.1322222	.2120212	.5599275	-.2501123	-.5181043	.8307917	2.194041	-.9800497
T	.0146543	.6559929	-.5296851	-.0594907	.0574219	2.570468	-2.075538	-.2331106
O	-.0179198	-.3026018	.9645613	-.0015981	-.0702174	-1.185726	3.779574	-.0062619

Example 4: Inequality contributions of sources of income with a nested structure

```
. shapowen ((Lh Lo) K)  ((P B) T) 0 , scalar(r(coeff)) : ///
>         sgini zero @ [aw=hpwgt]  , source
.....................................................
Instruction:  sgini zero @ [aw=hpwgt]  , source
Items list:   ((Lh Lo) K)  ((P B) T) 0
```

--- r(coeff) --- Empty set value: 0 Full set value: .2552037

	Nominal contribution				Relative contribution			
	Shapley -Owen	Banzhaf -Owen	First	Last	Shapley -Owen	Banzhaf -Owen	First	Last
FULL	.2552037				1			
(Lh Lo) K	8.289586	7.957234	.4881215	17.42046	32.48223	31.17993	1.912674	68.26099
Lh Lo	-1.810342	-2.278142	.4942429	-20.96295	-7.093712	-8.926759	1.93666	-82.14204
Lh	-.9820429	-1.216076	.5515417	-.3362227	-3.848074	-4.765118	2.161182	-1.317468
Lo	-.828299	-1.062066	.6249191	-.1139092	-3.245638	-4.161641	2.448707	-.4463463
K	10.09993	10.23538	.9316334	.0072015	39.57594	40.10669	3.650548	.0282187
(P B) T	-7.851159	-8.183512	-14.14202	-.2308853	-30.76428	-32.06658	-55.41464	-.9047097
P B	-3.700538	-3.866567	.5599275	-.2501123	-14.50033	-15.1509	2.194041	-.9800497
P	-1.921998	-2.005436	.8248165	-.1514365	-7.531229	-7.858175	3.231992	-.5933947
B	-1.77854	-1.861132	.6124187	-.0804831	-6.969099	-7.292729	2.399725	-.3153682
T	-4.150621	-4.316944	-.5296851	-.0594907	-16.26395	-16.91568	-2.075538	-.2331106
0	-.1832234	-.5155759	.9645613	-.0015981	-.7179494	-2.020252	3.779574	-.0062619

Example 4: Inequality contributions of sources of income: alternatives

- In the previous examples we created coalitions by “eliminating” income sources
- Alternatively one could
  - » eliminate *inequality in the source*: set it to its mean rather than to zero
  - » eliminate *association* of a source with any other income source
- `shapowen`’s `substitution(`*string*`)` option permits this: when a element is excluded from a coalition, it is *replaced* by the item in the same position in *string*

Example 5: Inequality contributions of sources of income: alternatives

Let's create some 'substitutes'

```
. foreach v of varlist Lh Lo K  PB T O {
  2.          su `v' [aw=hpwgt] , meanonly
  3.          gen mn_`v' = r(mean)
  4.          gen r_`v' = runiform()
  5.          egen rnd_`v' = clsort(`v' r_`v')  // ssc install _gclsort
  6. }
```

Example 5: Inequality contributions of sources of income: alternatives

```
. shapowen Lh Lo K  PB T O , scalar(r(coeff)) substitution(zero zero zero zero zero zero)    : ///
>         sgini @ [aw=hpwgt]  , source
.....................................................................
Instruction:  sgini @ [aw=hpwgt]  , source
Items list:   Lh Lo K  PB T O
```

--- r(coeff) --- Empty set value: 0 Full set value: .2552038

	Nominal contribution				Relative contribution			
	Shapley -Owen	Banzhaf -Owen	First	Last	Shapley -Owen	Banzhaf -Owen	First	Last
FULL	.2552038				1			
Lh	-.7171782	-.6947749	.5515405	-.3362227	-2.810217	-2.722431	2.161177	-1.317467
Lo	-.3581793	-.2311928	.6249173	-.1139092	-1.403503	-.9059146	2.448699	-.4463459
K	1.466175	2.491078	.9316009	.0072016	5.745116	9.761132	3.650419	.028219
PB	-.1322688	.2119414	.5599254	-.2501123	-.5182868	.830479	2.194032	-.9800491
T	.0146039	.6559077	-.5296869	-.0594906	.0572247	2.570133	-2.075545	-.2331103
O	-.0179493	-.3025758	.9642707	-.001598	-.0703331	-1.185624	3.778434	-.0062616

Example 5: Inequality contributions of sources of income: alternatives

```
. shapowen Lh Lo K  PB T O , scalar(r(coeff)) substitution(mn_Lh mn_Lo mn_K mn_PB mn_T mn_O) : ///
>         sgini @ [aw=hpwgt]  , source
.....................................................
Instruction:  sgini @ [aw=hpwgt]  , source
Items list:   Lh Lo K  PB T O
 ---  r(coeff) ---              Empty set value:        0                 Full set value:  .2552038
```

	Nominal contribution				Relative contribution			
	Shapley -Owen	Banzhaf -Owen	First	Last	Shapley -Owen	Banzhaf -Owen	First	Last
FULL	.2552038				1			
Lh	.1177067	.0989027	.3126045	-.0010125	.4612261	.387544	1.224921	-.0039675
Lo	.1123793	.093476	.2610494	.0402813	.4403512	.3662799	1.022906	.1578398
K	.0167581	.0134794	.0318558	.0156815	.0656654	.0528181	.1248249	.061447
PB	.0287063	.0186672	.1832561	-.08473	.1124839	.0731461	.7180774	-.3320091
T	-.0211016	-.0353484	.1853667	-.1696202	-.0826854	-.1385104	.7263477	-.6646459
O	.0007551	.0003568	.0038227	-.0005804	.0029589	.0013982	.0149791	-.0022742

Example 5: Inequality contributions of sources of income: alternatives

```
. shapowen Lh Lo K  PB T O , scalar(r(coeff)) substitution(rnd_Lh rnd_Lo rnd_K rnd_PB rnd_T rnd_O) : ///
>         sgini @ [aw=hpwgt]  , source
.....................................................
Instruction:  sgini @ [aw=hpwgt]  , source
Items list:   Lh Lo K  PB T O
```

--- r(coeff) --- Empty set value: .5878367 Full set value: .2552038

	Nominal contribution				Relative contribution			
	Shapley -Owen	Banzhaf -Owen	First	Last	Shapley -Owen	Banzhaf -Owen	First	Last
FULL	-.3326328				1			
Lh	-.1269625	-.1250364	-.0462109	-.2153681	.3816895	.3758993	.1389247	.6474648
Lo	-.0981312	-.0978587	-.063637	-.133664	.2950135	.2941944	.1913131	.4018365
K	.001636	.0015333	.0016604	.0020847	-.0049184	-.0046095	-.0049916	-.0062671
PB	-.0313957	-.0252985	.0431575	-.1302905	.0943855	.0760553	-.1297452	.3916945
T	-.0772337	-.0715256	.0248975	-.2021507	.2321889	.2150286	-.0748497	.6077293
O	-.0005458	-.0004681	-.0002925	-.0011227	.001641	.0014072	.0008793	.0033751

Example 6: Inequality contributions of population subgroups

- Say we partition the population in population groups – by age, education, hh type, etc.
- *What is the contribution of subgroups to aggregate inequality?*
- The Shapley value can be used here too (Deutsch and Silber, 2007, Giorgi and Guandalini, 2018)
- (illustration of `separator()` option to handle alternative list formats)

Example 6: Inequality contributions of population subgroups

```
. shapowen 1 2 3 4 5  , scalar(r(coeff))  sep(,) error(0)  : ///
>         sgini Yalt  [aw=hpwgt] if inlist(typehh2,@)
..Execution of --          sgini Yalt  [aw=hpwgt] if inlist(typehh2,) -- failed (error 198)
Set error value 0 used
.............................
Instruction:  sgini Yalt  [aw=hpwgt] if inlist(typehh2,@)
Items list:   1 2 3 4 5
```

--- r(coeff) --- Empty set value: 0 Full set value: .2552038

	Nominal contribution				Relative contribution			
	Shapley -Owen	Banzhaf -Owen	First	Last	Shapley -Owen	Banzhaf -Owen	First	Last
FULL	.2552038				1			
1	.0688676	.0356221	.2683823	.0124104	.2698532	.139583	1.051639	.0486294
2	.0680083	.0357219	.2697844	.0054535	.2664862	.1399741	1.057133	.021369
3	.0282575	-.0039953	.2198652	-.0242629	.1107253	-.0156554	.8615277	-.0950727
4	.0498535	.0184645	.2320633	.0032755	.1953479	.0723521	.9093254	.0128347
5	.0402169	.011902	.2045503	-.0007813	.1575873	.0466373	.8015174	-.0030613

Example 7: Decomposing the ‘composition effect’ in inequality trends

- Changes over time in the distribution of income can be driven by changes in returns to individual or hh characteristics and by changes in the composition of the population (again, by age, education, labour market participation, hh structures, etc.) .
- The ‘composition’ factor is a combination of changes in multiple characteristics
- *What is the contribution of changes in different characteristics to inequality changes?*
- (illustration of use of small `program define` wrappers)

Example 7: Decomposing the ‘composition effect’ in inequality trends

```
. cap pr drop changini
. pr def changini , rclass
  1.          tempvar p rw
  2.          svy : logit period ‘0’
  3.          qui predict ‘p’ , rules
  4.          qui gen ‘rw’ = cond(period==1 , 1 , ‘p’/(1-‘p’))
  5.          sgini Yalt [aw=hpwgt*‘rw’] if period==0
  6.          return scalar gini = r(coeff)
  7. end
```

Example 7: Decomposing the ‘composition effect’ in inequality trends

```
. sgini Yalt [aw=hpwgt] if period==0
Gini coefficient for Yalt
```

Variable	v=2
Yalt	0.2715

```
. sgini Yalt [aw=hpwgt] if period==1
Gini coefficient for Yalt
```

Variable	v=2
Yalt	0.2552

Example 7: Decomposing the ‘composition effect’ in inequality trends

```
. changini i.typehh2 c.shearn i.educ i.sex
(running logit on estimation sample)
Survey: Logistic regression

Number of strata   =         2                  Number of obs     =      10,889
Number of PSUs     =    10,889                  Population size   =  20,475,368
                                                Design df         =      10,887
                                                F(   8,  10880)   =      179.16
                                                Prob > F          =      0.0000
```

		Linearized				
	period	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
typehh2						
	Couple only	.2168325	.08322	2.61	0.009	.0537062	.3799587
Couple with child(ren) and possibly others		-.1628252	.0799651	-2.04	0.042	-.3195713	-.006079
Single with child(ren) and possibly others		-.7765123	.1326156	-5.86	0.000	-1.036463	-.5165617
	Other configuration	-.0360524	.315691	-0.11	0.909	-.6548643	.5827594
	shearn	1.145561	.0851836	13.45	0.000	.9785854	1.312536
educ							
	[2]medium	1.217598	.0744029	16.36	0.000	1.071754	1.363441
	[3]high	1.599592	.0726049	22.03	0.000	1.457273	1.741911
sex							
	[2]female	1.877692	.0671318	27.97	0.000	1.746101	2.009282
	_cons	-1.619951	.078148	-20.73	0.000	-1.773135	-1.466767

Gini coefficient for Yalt

Variable	v=2
Yalt	0.2521

Example 7: Decomposing the ‘composition effect’ in inequality trends

```
. shapowen i.typehh2 c.shearn i.educ i.sex , scal(r(gini))   : ///
>         changini @
.................
Instruction:  changini @
Items list:   i.typehh2 c.shearn i.educ i.sex
```

--- r(gini) --- Empty set value: .2714685 Full set value: .2521056

	Nominal contribution				Relative contribution			
	Shapley -Owen	Banzhaf -Owen	First	Last	Shapley -Owen	Banzhaf -Owen	First	Last
FULL	-.019363				1			
i.typehh2	.0008653	.0012075	.001221	-.000859	-.0446902	-.0623614	-.0630602	.0443643
c.shearn	-.0191425	-.0186721	-.0211986	-.0189679	.9886135	.9643199	1.094803	.9795988
i.educ	-.0032951	-.0028207	-.0050923	-.0033955	.1701762	.1456759	.2629924	.1753611
i.sex	.0022093	.0026497	.0024182	.0002386	-.1140995	-.1368462	-.12489	-.0123222

Example 7: Decomposing the ‘composition effect’ in inequality trends (bootstrap inference)

- Prefix commands can be combined
- Earlier examples show that `svy:` can seamlessly be used in the *cmd ... @ ...* pseudo instruction
- The `bootstrap` prefix can be used with `shapowen`

Example 7: Decomposing the ‘composition effect’ in inequality trends (bootstrap inference)

```
.         bootstrap ///
>                 a=el(r(ShapOw),2,1)  ///
>                 b=el(r(ShapOw),3,1)  ///
>                 c=el(r(ShapOw),4,1)  ///
>                 d=el(r(ShapOw),5,1)  ///
>                 , reps(499) : ///
>                 shapowen i.typehh2 c.shearn i.educ i.sex , scal(r(gini))   : ///
>                         changini @
(running shapowen on estimation sample)

Bootstrap replications (499)
----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
..................................................    50
<SNIP>
.............................................
```

Example 7: Decomposing the ‘composition effect’ in inequality trends (bootstrap inference)

```
Bootstrap results                                   Number of obs      =     10,889
                                                    Replications       =        499

      command:  shapowen i.typehh2 c.shearn i.educ i.sex, scal(r(gini)) : changini @
            a:  el(r(ShapOw),2,1)
            b:  el(r(ShapOw),3,1)
            c:  el(r(ShapOw),4,1)
            d:  el(r(ShapOw),5,1)
```

	Observed Coef.	Bootstrap Std. Err.	z	P>|z|	Normal-based [95% Conf.	Interval]
a	.0008653	.0008148	1.06	0.288	-.0007317	.0024624
b	-.0191425	.002364	-8.10	0.000	-.0237759	-.0145091
c	-.0032951	.0023183	-1.42	0.155	-.0078389	.0012487
d	.0022093	.0021852	1.01	0.312	-.0020737	.0064923

Example 8: Multidimensional index decomposition

- We have a 'multidimensional' index of deprivation $P(\mathbf{d})$ where $\mathbf{d}$ is a vector of binary items of deprivation ("not being able to face unexpected expenses", "not being able to go on vacation", "not being able to replace worn out furniture", etc.)
- Many such measures exist.
- Many such measures are non-linear in the elements of $\mathbf{d}$. E.g., Aaberge et al. (2019) propose a measure that is sensitive to the concentration of multiple items on a small set of individuals

What is the contribution of specific items to the aggregate poverty measure?

Example 8: Multidimensional index decomposition

```
. cap pr drop __aps

. pr def __aps , rclass
  1.         syntax [varlist(default=none)] [fw aw pw iw] [, param(real 2)]
  2.         tempvar tot mtot
  3.         qui egen `tot' = rowtotal(`varlist')
  4.         qui gen `mtot' = -`tot'
  5.         sgini `tot' [`weight'`exp']  , param(`param') welfare sort(`mtot')
  6.         return scalar coeff = r(coeff)
  7. end
```

Example 8: Multidimensional index decomposition

```
. shapowen ///
>         ad_cloths ad_shoes ad_frien ad_leis ad_pock ad_internet h_furniture h_car h_warm h_holid h_unex h_meat h_arrears ///
>         , scalar(r(coeff)) emptyval(0) nodots approximation(200) : ///
>         __aps @ [pw=pwgt]

Instruction:  __aps @ [pw=pwgt]
Items list:   ad_cloths ad_shoes ad_frien ad_leis ad_pock ad_internet h_furniture h_car h_warm h_holid h_unex h_meat h_arrears
Number of random permutations:    200
Number of evaluations required:   1556
 ---  r(coeff) ---              Empty set value:        0                    Full set value:  2.21694
```

		Nominal contribution					Relative contribution				
		Shapley -Owen	Banzhaf	First	Last	Seq.	Shapley -Owen	Banzhaf	First	Last	Seq.
[0]	FULL	2.21694					1				
[1]	ad_cloths	.147857	.	.153658	.146973	.153658	.066694	.	.069311	.066296	.069311
[1]	ad_shoes	.048597	.	.050695	.048355	.049994	.021921	.	.022867	.021812	.022551
[1]	ad_frien	.096711	.	.101471	.096238	.098693	.043624	.	.045771	.04341	.044518
[1]	ad_leis	.209352	.	.219042	.207288	.214497	.094433	.	.098804	.093502	.096754
[1]	ad_pock	.170595	.	.179642	.168521	.173893	.076951	.	.081032	.076015	.078438
[1]	ad_internet	.020703	.	.022033	.02044	.020845	.009338	.	.009938	.00922	.009402
[1]	h_furniture	.262123	.	.27557	.259021	.262871	.118236	.	.124302	.116837	.118574
[1]	h_car	.0443	.	.04814	.043199	.043724	.019982	.	.021715	.019486	.019723
[1]	h_warm	.189182	.	.206334	.182879	.186675	.085335	.	.093072	.082492	.084204
[1]	h_holid	.337327	.	.355694	.331695	.333558	.152159	.	.160444	.149619	.150459
[1]	h_unex	.398259	.	.414002	.392558	.392379	.179644	.	.186745	.177072	.176992
[1]	h_meat	.157563	.	.168665	.155336	.155775	.071072	.	.07608	.070068	.070266
[1]	h_arrears	.134369	.	.145813	.130376	.130376	.06061	.	.065772	.058809	.058809

Example 8: Multidimensional index decomposition

```
. shapowen ///
>         (ad_cloths ad_shoes h_meat h_warm ) (ad_internet h_furniture h_car)  (ad_frien ad_leis h_holid)  (ad_pock  h_unex  h_arrears) ///
>         , scalar(r(coeff)) emptyval(0) nodots  approximation(200) :    ///
>         __aps @ [pw=pwgt]

Instruction:  __aps @ [pw=pwgt]
Items list:   (ad_cloths ad_shoes h_meat h_warm ) (ad_internet h_furniture h_car)  (ad_frien ad_leis h_holid)  (ad_pock  h_unex  h_arrears)
Number of random permutations:    200
Number of evaluations required:   271
 ---  r(coeff) ---              Empty set value:        0                    Full set value:  2.21694
```

		Nominal contribution					Relative contribution				
		Shapley -Owen	Banzhaf	First	Last	Seq.	Shapley -Owen	Banzhaf	First	Last	Seq.
[0]	FULL	2.21694					1				
[1]	ad_cloths ad_shoes h_m~m	.542648	.	.561748	.532734	.561748	.244774	.	.253389	.240302	.253389
[2]	ad_cloths	.147399	.	.153658	.146973	.153658	.066488	.	.069311	.066296	.069311
[2]	ad_shoes	.048532	.	.050695	.048355	.049994	.021892	.	.022867	.021812	.022551
[2]	h_meat	.157931	.	.168665	.155336	.163056	.071238	.	.07608	.070068	.07355
[2]	h_warm	.188786	.	.206334	.182879	.195041	.085156	.	.093072	.082492	.087978
[1]	ad_internet h_furnitur~r	.328152	.	.341946	.322602	.325673	.14802	.	.154242	.145517	.146902
[2]	ad_internet	.020679	.	.022033	.02044	.020435	.009328	.	.009938	.00922	.009218
[2]	h_furniture	.263262	.	.27557	.259021	.261714	.11875	.	.124302	.116837	.118052
[2]	h_car	.04421	.	.04814	.043199	.043524	.019942	.	.021715	.019486	.019633
[1]	ad_frien ad_leis h_holid	.643372	.	.662094	.635107	.638321	.290207	.	.298653	.28648	.287929
[2]	ad_frien	.096829	.	.101471	.096238	.095928	.043677	.	.045771	.04341	.043271
[2]	ad_leis	.20915	.	.219042	.207288	.208148	.094342	.	.098804	.093502	.09389
[2]	h_holid	.337393	.	.355694	.331695	.334245	.152189	.	.160444	.149619	.150769
[1]	ad_pock h_unex h_arrs	.702766	.	.722446	.691196	.691196	.316999	.	.325876	.31178	.31178

Example 8: Multidimensional index decomposition

```
. shapowen ///
>         (ad_cloths ad_shoes h_meat h_warm ) (ad_internet h_furniture h_car)  (ad_frien ad_leis h_holid)  (ad_pock  h_unex  h_arrears)  ///
>         , scalar(r(coeff))  emptyval(0)  nodots  approximation(200) :    ///
>         __aps @ [pw=pwgt] , param(3)

Instruction:  __aps @ [pw=pwgt] , param(3)
Items list:   (ad_cloths ad_shoes h_meat h_warm ) (ad_internet h_furniture h_car)  (ad_frien ad_leis h_holid)  (ad_pock  h_unex  h_arrears)
Number of random permutations:    200
Number of evaluations required:   267
 ---  r(coeff) ---              Empty set value:        0                    Full set value:  3.03841
```

		Nominal contribution					Relative contribution				
		Shapley -Owen	Banzhaf	First	Last	Seq.	Shapley -Owen	Banzhaf	First	Last	Seq.
[0]	FULL	3.03841					1				
[1]	ad_cloths ad_shoes h_m~m	.754116	.	.823154	.722531	.823154	.248194	.	.270916	.237799	.270916
[2]	ad_cloths	.211205	.	.236852	.207916	.236852	.069512	.	.077953	.068429	.077953
[2]	ad_shoes	.071404	.	.081004	.070007	.077737	.023501	.	.02666	.023041	.025585
[2]	h_meat	.219433	.	.258571	.21101	.237348	.07222	.	.085101	.069447	.078116
[2]	h_warm	.252074	.	.311939	.235174	.271217	.082962	.	.102665	.0774	.089263
[1]	ad_internet h_furnitur~r	.454399	.	.503697	.435906	.445932	.149552	.	.165776	.143465	.146765
[2]	ad_internet	.030176	.	.035545	.02902	.029473	.009932	.	.011698	.009551	.0097
[2]	h_furniture	.36185	.	.405657	.347904	.356124	.119092	.	.13351	.114502	.117207
[2]	h_car	.062373	.	.076988	.059081	.060335	.020528	.	.025338	.019445	.019858
[1]	ad_frien ad_leis h_holid	.882476	.	.943366	.850927	.860097	.29044	.	.31048	.280056	.283074
[2]	ad_frien	.140405	.	.159336	.136956	.137502	.04621	.	.052441	.045075	.045255
[2]	ad_leis	.294728	.	.329569	.285106	.288772	.097001	.	.108467	.093834	.095041
[2]	h_holid	.447344	.	.506794	.428328	.433822	.147229	.	.166796	.140971	.142779
[1]	ad_pock h_unex h_arrs	.947421	.	1.01324	.90923	.90923	.311814	.	.333476	.299245	.299245

Example 8: Multidimensional index decomposition

```
. shapowen ///
>         "ad_cloths ad_shoes h_meat h_warm" "ad_internet h_furniture h_car"  "ad_frien ad_leis h_holid"  "ad_pock  h_unex  h_arrears" ///
>         , scalar(r(coeff))  emptyval(0)  :    ///
>         __aps @ [pw=pwgt] , param(3)
.................
Instruction:  __aps @ [pw=pwgt] , param(3)
Items list:   "ad_cloths ad_shoes h_meat h_warm" "ad_internet h_furniture h_car"  "ad_frien ad_leis h_holid"  "ad_pock  h_unex  h_arrears"
Number of evaluations required:  16
 ---  r(coeff) ---              Empty set value:        0                    Full set value:  3.03841
```

		Nominal contribution					Relative contribution				
		Shapley -Owen	Banzhaf	First	Last	Seq.	Shapley -Owen	Banzhaf	First	Last	Seq.
[0]	FULL	3.03841					1				
[1]	"ad_cloths ad_shoes h_~r	.756646	.748548	.823154	.722531	.823154	.249027	.246362	.270916	.237799	.270916
[1]	"ad_internet h_furnitu~"	.45598	.449069	.503697	.435906	.445932	.150072	.147797	.165776	.143465	.146765
[1]	"ad_frien ad_leis h_ho~"	.881129	.87312	.943366	.850927	.860097	.289996	.28736	.31048	.280056	.283074
[1]	"ad_pock h_unex h_ar~"	.944659	.936371	1.01324	.90923	.90923	.310905	.308178	.333476	.299245	.299245

Closing

- The Shapley value and Shapley value decompositions and their Owen counterparts for nested structures have many potential applications
- `shapowen` (available on SSC) facilitates their calculation with a generic prefix-based syntax and flexible input processing

Comments, feedback and suggestions welcome!

References i

Chantreuil, F., Courtin, S., Fourrey, K. and Lebon, I. (2019), 'A note on the decomposability of inequality measures', *Social Choice and Welfare* **53**(2), 283–298.

Chantreuil, F. and Trannoy, A. (2013), 'Inequality decomposition values: the trade-off between marginality and efficiency', *Journal of Economic Inequality* **11**(1), 83–98.

Deutsch, J. and Silber, J. (2007), Decomposing income inequality by population subgroups: A generalization, *in* 'Inequality and Poverty', Research on Economic Inequality, Emerald Group Publishing Limited, pp. 237–253.

Giorgi, G. M. and Guandalini, A. (2018), 'Decomposing the Bonferroni inequality index by subgroups: Shapley value and balance of inequality', *Econometrics* **6**(2).

Kolenikov, S. (2000), 'Shapley: Stata module to perform additive decomposition of sample statistic'. **URL:** *https://EconPapers.repec.org/RePEc:boc:bocode:s411401*

Owen, G. (1977), Values of games with a priori unions, *in* R. Henn and O. Moeschlin, eds, 'Mathematical Economics and Game Theory', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 76–88.

References ii

Sastre, M. and Trannoy, A. (2002), 'Shapley inequality decomposition by factor components: Some methodological issues', *Journal of Economics* **77**(1), 51–89.
URL: *http://dx.doi.org/10.1007/BF03052500*

Shapley, L. S. (1953), A value for n-person games, *in* H. Kuhn and A. W. Tucker, eds, 'Contributions to the Theory of Games, Vol. II', Princeton: Princeton University Press, Princeton, NJ, pp. 307–317.

Shorrocks, A. F. (2013), 'Decomposition procedures for distributional analysis: a unified framework based on the Shapley value', *Journal of Economic Inequality* **11**(1), 99–126.
URL: *http://dx.doi.org/10.1007/s10888-011-9214-z*

Trannoy, A. and Sastre, M. (2002), Shapley inequality decomposition by factor components: Some methodological issues, *in* P. Moyes, C. Seidl and A. F. Shorrocks, eds, 'Inequalities: Theory, Measurement and Applications', number 9 *in* 'Journal of Economics Supplement Issues', Springer-Verlag, Wien, pp. 51–89.