Shapley Effects for Use as Sensitivity Measure

Elmar Plischke

Institut für Ressourcenökologie
Helmholtz-Zentrum Dresden-Rossendorf

SAMO Summer School, Parma, June 2024

Contents

Motivation

For independent variables

$$\mathbb{V}[X_1 + X_2] = \mathbb{V}[X_1] + \mathbb{V}[X_2]$$

but generally

$$\mathbb{V}[X_1 + X_2] = \mathbb{E}[(X_1 + X_2)^2] - (\mathbb{E}[X_1 + X_2])^2 = \mathbb{E}[X_1^2] - \mathbb{E}[X_1]^2 + \mathbb{E}[X_2^2] - \mathbb{E}[X_2]^2 \\ + 2\,(\mathbb{E}[X_1 X_2] - \mathbb{E}[X_1]\,\mathbb{E}[X_2]) = \mathbb{V}[X_1] + \mathbb{V}[X_2] + 2\,\mathrm{Cov}(X_1, X_2)$$

Motivation

For independent variables

$$\mathbb{V}[X_1 + X_2] = \mathbb{V}[X_1] + \mathbb{V}[X_2]$$

but generally

$$\mathbb{V}[X_1 + X_2] = \mathbb{E}[(X_1 + X_2)^2] - (\mathbb{E}[X_1 + X_2])^2 = \mathbb{E}[X_1^2] - \mathbb{E}[X_1]^2 + \mathbb{E}[X_2^2] - \mathbb{E}[X_2]^2 \\ + 2\,(\mathbb{E}[X_1 X_2] - \mathbb{E}[X_1]\,\mathbb{E}[X_2]) = \mathbb{V}[X_1] + \mathbb{V}[X_2] + 2\,\mathrm{Cov}(X_1, X_2)$$

Consequences for functional ANOVA under dependence

- Covariance terms need to be considered
- Orthogonality (strong annihilation) is lost: Hierarchical orthogonality can be used
- But this introduces dependence on the order of the factors in the model

First and last term in any ordering of the factors may receive special attention

Wanted

A concept to define main and total effects and related sensitivity indices without recurring to functional ANOVA decomposition

Back to the basics:

- Main effect S_i: Variance explained by a functional dependence on X_i

Wanted

A concept to define main and total effects and related sensitivity indices without recurring to functional ANOVA decomposition

Back to the basics:

- Main effect S_i: Variance explained by a functional dependence on X_i
- Total effect T_i: Residual variance un-explained by a functional dependence on X_{-i}

Wanted

A concept to define main and total effects and related sensitivity indices without recurring to functional ANOVA decomposition

Back to the basics:

- Main effect S_i: Variance explained by a functional dependence on X_i
- Total effect T_i: Residual variance un-explained by a functional dependence on X_{-i}

We study a game theoretic approach:
The goal is to attribute a fair share of the variance to each input factor

Game Theory: Definitions

For d players,

- Coalition-worth value function $\mathrm{val} : 2^d \to \mathbb{R}_{\geq 0}$, 2^d: set of subsets of $[d] := \{1, \ldots, d\}$
- Coalition $\alpha \subset [d]$ lists the active players, anti-coalition $\sim \alpha = [d] \setminus \alpha$
- Marginal contribution of player i joining coalition α: $\mathrm{mar}(\alpha, i) = \mathrm{val}(\alpha \cup \{i\}) - \mathrm{val}(\alpha)$

The value function assigns a payoff to a group of players
The value function is a game if it is grounded: $\mathrm{val}(\emptyset) = 0$.

Grand total: $\mathrm{val}([d])$

Axioms for the Shapley Value

Attribute a fair (egalitarian) share of the grand total to each player:

Theorem

The Shapley value $\Phi_i(\mathrm{val})$ *of player* i *for the payoffs* val *is uniquely characterized by the following four axioms,*

- *Pareto-efficiency:* $\sum_{i=1}^{d} \Phi_i(\mathrm{val}) = \mathrm{val}([d])$
- *Symmetry: If* $\mathrm{val}(\alpha \cup \{i\}) = \mathrm{val}(\alpha \cup \{j\})$ *for all subsets* α *containing neither* i *nor* j *then* $\Phi_i(\mathrm{val}) = \Phi_j(\mathrm{val})$
- *Linearity:* $\Phi_i(\mathrm{val}_1 + \mathrm{val}_2) = \Phi_i(\mathrm{val}_1) + \Phi_i(\mathrm{val}_2)$
- *Null-player: If for all* α, $\mathrm{val}(\alpha \cup \{i\}) = \mathrm{val}(\alpha)$ *holds then* $\Phi_i(\mathrm{val}) = 0$.

Formulas for the Shapley Value

$$\Phi_i(\mathrm{val}) = \frac{1}{d} \sum_{\alpha : i \notin \alpha} \binom{d-1}{|\alpha|}^{-1} \mathrm{mar}(\alpha, i)$$

$$\Phi_i(\mathrm{val}) = \frac{1}{d} \sum_{\alpha : i \in \alpha} \binom{d-1}{|\alpha|-1}^{-1} (\mathrm{val}(\alpha) - \mathrm{val}(\sim \alpha))$$

$$\Phi_i(\mathrm{val}) = \sum_{\alpha : i \in \alpha} \frac{\mathrm{mob}(\alpha)}{|\alpha|}$$

All three formulas satisfy the axioms which uniquely describe the Shapley value, hence define the same object.

Möbius inverses

Unique decomposition $\mathrm{val}(\alpha) = \sum_\beta \mathrm{mob}(\beta) u_\beta(\alpha)$
$u_\beta(\alpha) = \mathbf{1}(\beta \subset \alpha)$ (Unanimity game) codes subset inclusion
Weights: Möbius inverses / Harsanyi dividends. Implicitly defined by

$$\mathrm{val}(\alpha) = \sum_{\beta \subset \alpha} \mathrm{mob}(\beta).$$

This system of $2^d - 1$ linear equations can be solved by an inclusion-exclusion rule

$$\mathrm{mob}(\alpha) = \sum_{\beta \subset \alpha} (-1)^{|\alpha|+|\beta|} \mathrm{val}(\beta).$$

This approach is technical equivalent to the formation of higher order effects [Plischke et al., 2021]

Main and total effects for games

Let us therefore introduce (unnormalized) main and total effects based on the coalition-worth value function,

- Main effects $S_i = \text{val}(\{i\}) = \text{mar}(\emptyset, i) = \text{mob}(\{i\})$
- Total effects $T_i = \sum_{\alpha : i \in \alpha} \text{mob}(\alpha)$

Note that always

$$T_i = \sum_{\alpha : i \in \alpha} \text{mob}(\alpha) = \sum_{\alpha} \text{mob}(\alpha) - \sum_{\alpha : i \notin \alpha} \text{mob}(\alpha)$$

$$= \text{val}([d]) - \sum_{\alpha \subset \sim i} \text{mob}(\alpha) = \text{val}([d]) - \text{val}(\sim i)$$

Shapley Effects

Grand total: Output variance
Players: Input factors

Consider the value function $\mathrm{val}(\alpha) = \mathbb{V}[\mathbb{E}[Y|X_\alpha]]$:
If $\alpha = \emptyset$ then we compute the variance of a constant value, i.e. val is a game
If $\alpha = [d]$ and $y = f(x_1, \ldots, x_d)$ is a square integrable deterministic function then $\mathrm{val}([d]) = \mathbb{V}[\mathbb{E}[Y|X_{[d]}]] = \mathbb{V}[Y]$, i.e. the grand total is the output variance

How to compute the Shapley effects

Sobol' method, pick-and-freeze with conditionally independent sampling

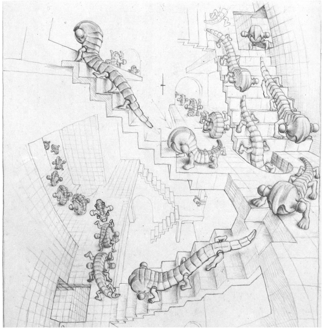
```
for i=1:d; w0=1; for j=1:i-1; w0=w0*(d-i+j)/j; end; w(i)=w0; end % weights
[ua,ub]=createsample(d,n,randomsource); za=norminv(ua); zb=norminv(ub);
C=chol(S); na=za*C; nb=zb*C; xa=trafo(normcdf(na)); xb=trafo(normcdf(nb));
ya=model(xa);yb=model(xb); Vy=(yb-ya)'*(yb-ya)/n/2; Shap=ones(1,d)*Vy;
for i=1:2^(d-1)-1 % loop only over half of the indices
 g=logical(bitget(i,1:d)); sz=sum(g); D=chol([S(g,g),S(g,¬g);S(¬g,g),S(¬g,¬g)]);
 D11=D(sz+1:end,sz+1:end); D22=D(1:sz,1:sz);D21=D(1:sz,sz+1:end);
 ni=na;ni(:,¬g)=zb(:,¬g)*D11+na(:,g)*(D22\D21); xi=trafo(normcdf(ni));
 yi=model(xi); sz=k-sz; E=chol([S(¬g,¬g),S(¬g,g);S(g,¬g),S(g,g)]);
 E11=E(sz+1:end,sz+1:end); E22=E(1:sz,1:sz); E21=E(1:sz,sz+1:end); nj=na;
 nj(:,g)=zb(:,g)*E11+na(:,¬g)*(E22\E21);  xj=trafo(normcdf(nj));
 yj=model(xj); sz=k-sz; bal=(yj-yi)'*(yj+yi-2*ya)/(2*n); % bal =val(g)-val(¬g)
 Shap(g)=Shap(g)+bal/w(sz); Shap(¬g)=Shap(¬g)-bal/w(d-sz);
end, Shap=Shap/d;
```

Code Discussion

- `d` input dimension, `n` basic sample block size, `model` vectorized simulator, `trafo` marginal transformation from $[0,1]^d$, `createsample` create two basic sample blocks (not shown)
- Implemented are Gaussian Copula dependence structures
- Via Cholesky decompositions of reordered covariance matrices
- Using the second Shapley formula with a balanced value function
- Computationally costly: Visits half of all subsets, pick-and-freeze design for each of them, symmetric design

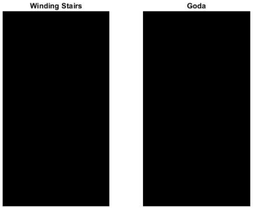
Goda "Crawling Centipede" approach

With a winding stairs approach one can compute val(α) for $\alpha = \{1, 2, \ldots, i\}$ (consecutive indices).
Goda's approach [Goda, 2021]: Randomize the index were the innovation enters



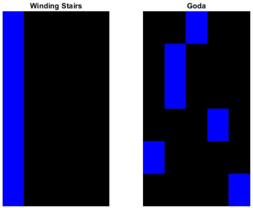
Goda “Crawling Centipede” approach

With a winding stairs approach one can compute val(α) for $\alpha = \{1, 2, \ldots, i\}$ (consecutive indices).
Goda’s approach [Goda, 2021]: Randomize the index were the innovation enters



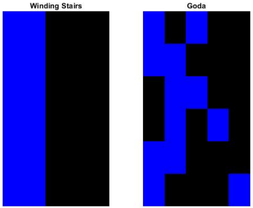
Goda “Crawling Centipede” approach

With a winding stairs approach one can compute val(α) for $\alpha = \{1, 2, \ldots, i\}$ (consecutive indices).
Goda’s approach [Goda, 2021]: Randomize the index were the innovation enters



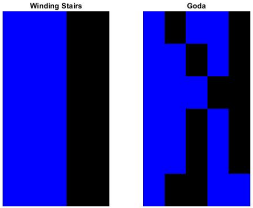
Goda "Crawling Centipede" approach

With a winding stairs approach one can compute val(α) for $\alpha = \{1, 2, \ldots, i\}$ (consecutive indices).
Goda's approach [Goda, 2021]: Randomize the index were the innovation enters



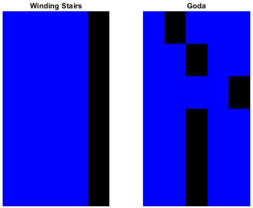
Goda "Crawling Centipede" approach

With a winding stairs approach one can compute val(α) for $\alpha = \{1, 2, \ldots, i\}$ (consecutive indices).
Goda's approach [Goda, 2021]: Randomize the index were the innovation enters



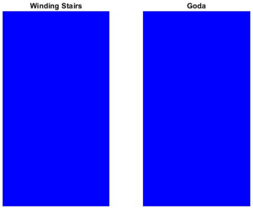
Goda "Crawling Centipede" approach

With a winding stairs approach one can compute val(α) for $\alpha = \{1, 2, \ldots, i\}$ (consecutive indices).
Goda's approach [Goda, 2021]: Randomize the index were the innovation enters



Goda "Crawling Centipede" approach

With a winding stairs approach one can compute val(α) for $\alpha = \{1, 2, \ldots, i\}$ (consecutive indices).
Goda's approach [Goda, 2021]: Randomize the index were the innovation enters



Shapley effects, Goda's method

```
x = rand(n,d);  y = rand(n,d); % MC sample
[¬, pm] = sort(rand(n,d),2);  % random permutation
z = x; fz1 = func(trafo(z)); fx = fz1; % save fx as reference point
phi1 = zeros(1,d); phi2 = zeros(1,d);
for j=1:d
   % activate indices from permutation matrix
   ind = bsxfun(@eq,pm(:,j),1:d); % compare column with row
   z(ind) = y(ind); % copy over next pick'freeze dimension (per run)
   fz2 = func(trafo(z));
   fmarg = ((fx-fz1/2-fz2/2).*(fz1-fz2))';  % update
   phi1 = phi1 + fmarg*ind/n;
   fz1 = fz2;
end
```

Code Discussion

- `d` input dimension, `n` basic sample block size, `func` vectorized simulator, `trafo` marginal transformation from $[0,1]^d$
- Only input independence (1D innovation injection)
- Reference point is the $f(x)$ output, but may also consider differences to $f(y)$
- Original version offers error estimate
- Computationally cheap: $(d+1)\cdot n$ vs. $(2^d-1)\cdot n$

Analytical Example: Gauss Linear

Input:	Multivariate normal distribution with covariance Σ
Simulation model:	$Y = \beta^0 + \beta^T X$, $X \in \mathbb{R}^d$

All conditional distributions are Gaussian, all conditional expectations are linear

Analytical Example: Gauss Linear

Input: Multivariate normal distribution with covariance Σ

Simulation model: $Y = \beta^0 + \beta^T X,\ X \in \mathbb{R}^d$

All conditional distributions are Gaussian, all conditional expectations are linear

Theorem

Under Gauss linear, unnormalized main, total and Shapley effects are given by

$$S_j = \beta^T \left(\frac{\Sigma_{[d],j}\Sigma_{j,[d]}}{\Sigma_{j,j}} \right) \beta = \beta^T \left(\frac{\Sigma_{[d],j}\Sigma_{[d],j}^T}{\Sigma_{j,j}} \right) \beta$$

$$T_j = \beta_j^2 \frac{\det(\Sigma)}{\det(\Sigma_{-j,-j})}$$

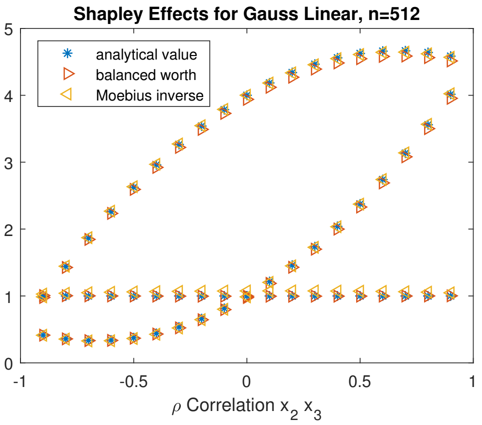
$$\Phi_j = \frac{1}{d} \sum_{j \in u} \binom{d-1}{|u|-1}^{-1} \beta^T \left(\Sigma_{[d],u}\Sigma_{u,u}^{-1}\Sigma_{u,[d]} - \Sigma_{[d],-u}\Sigma_{-u,-u}^{-1}\Sigma_{-u,[d]} \right) \beta$$

Output variance is $\mathbb{V}[Y] = \beta^T \Sigma \beta$.

Feed the Code

Input: $X \sim \mathcal{N}(0, \Sigma)$ with $\Sigma = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & \varrho\sigma \\ 0 & \varrho\sigma & \sigma^2 \end{pmatrix}$, $\sigma = 2$, ϱ is varied within $[-1, 1]$

Model $Y = f(X_1, X_2, X_3) = X_1 + X_2 + X_3$



Computing Shapley values, II

Using $\mathrm{Sh}_i = \sum_{\alpha : i \in \alpha} \frac{\mathrm{mob}(\alpha)}{|\alpha|}$ [Grabisch, 2006, Owen, 2014]: fast Möbius inverse needed

Computing Shapley values, II

Using $\mathrm{Sh}_i = \sum_{\alpha : i \in \alpha} \frac{\mathrm{mob}(\alpha)}{|\alpha|}$ [Grabisch, 2006, Owen, 2014]: fast Möbius inverse needed
Fast multiplication algorithms [Yates, 1937, Good, 1958]: iterated Kronecker products
$A \otimes^d A = (A \otimes^{d-1} A) \otimes (A \otimes^{d-1} A)$ with $A \otimes^0 A = A$.

Theorem

Let $v = (\mathrm{val}(\emptyset), \mathrm{val}(\{1\}), \mathrm{val}(\{2\}), \mathrm{val}(\{1,2\}), \mathrm{val}(\{3\}), \ldots, \mathrm{val}(\{1, \ldots, d\}))^T$ be a 2^d vector in natural order (binary coded). Möbius inverse is obtained by left-multiplication with the iterated Kronecker product of $A = \left(\begin{smallmatrix} 1 & 0 \\ -1 & 1 \end{smallmatrix}\right)$.

$A^{-1} = \left(\begin{smallmatrix} 1 & 0 \\ 1 & 1 \end{smallmatrix}\right)$ codes inclusion in 2^1: $\emptyset \subset \emptyset$, $\emptyset \subset \alpha$, $\alpha \subset \alpha$.

Illustrating Fast Möbius/Yates Transformation

$$(\text{first,last}) \cdots \mapsto (\text{all first, all last-first})$$

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
0	31	44	75	0	56	44	100	val α
								Step 1
								Step 2
								mob(α)
								$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$								Sh_1
$\sum$								Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

(first,last) $\cdots \mapsto$ (all first, all last-first)

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
								Step 1
								Step 2
								mob(α)
								$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$								Sh_1
$\sum$								Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

(first,last) $\cdots \mapsto$ (all first, all last-first)

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
(0	44	0	44)					Step 1
								Step 2
								mob(α)
								$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$								Sh_1
$\sum$								Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

(first,last) $\cdots \mapsto$ (all first, all last-first)

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
(0	44	0	44)	(31	31	56	56)	Step 1
								Step 2
								mob(α)
								$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$								Sh_1
$\sum$								Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

(first,last) $\cdots \mapsto$ (all first, all last-first)

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
(0	44)	(0	44)	(31	31)	(56	56)	Step 1
								Step 2
								mob(α)
								$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$								Sh_1
$\sum$								Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

(first,last) $\cdots \mapsto$ (all first, all last-first)

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
(0	44)	(0	44)	(31	31)	(56	56)	Step 1
(0	0	31	56)					Step 2
								mob(α)
								$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$								Sh_1
$\sum$								Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

(first,last) $\cdots \mapsto$ (all first, all last-first)

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
(0	44)	(0	44)	(31	31)	(56	56)	Step 1
(0	0	31	56)	(44	44	0	0)	Step 2
								mob(α)
								$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$								Sh_1
$\sum$								Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

$$(\text{first,last}) \cdots \mapsto (\text{all first, all last-first})$$

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
(0	44)	(0	44)	(31	31)	(56	56)	Step 1
(0	0)	(31	56)	(44	44)	(0	0)	Step 2
								mob(α)
								$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$								Sh_1
$\sum$								Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

(first,last) $\cdots \mapsto$ (all first, all last-first)

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
(0	44)	(0	44)	(31	31)	(56	56)	Step 1
(0	0)	(31	56)	(44	44)	(0	0)	Step 2
(0	31	44	0)					mob(α)
								$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$								Sh_1
$\sum$								Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

$$\text{(first,last)} \cdots \mapsto \text{(all first, all last-first)}$$

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
(0	44)	(0	44)	(31	31)	(56	56)	Step 1
(0	0)	(31	56)	(44	44)	(0	0)	Step 2
(0	31	44	0)	(0	25	0	0)	mob(α)
								$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$								Sh_1
$\sum$								Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

(first,last) $\cdots \mapsto$ (all first, all last-first)

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
(0	44)	(0	44)	(31	31)	(56	56)	Step 1
(0	0)	(31	56)	(44	44)	(0	0)	Step 2
0	31	44	0	0	25	0	0	mob(α)
	31	44	0	0	12.5	0	0	$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$								Sh_1
$\sum$								Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

(first,last) $\cdots \mapsto$ (all first, all last-first)

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
(0	44)	(0	44)	(31	31)	(56	56)	Step 1
(0	0)	(31	56)	(44	44)	(0	0)	Step 2
0	31	44	0	0	25	0	0	mob(α)
	31	44	0	0	12.5	0	0	$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$	31		0		12.5		0	Sh_1
$\sum$								Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

(first,last) $\cdots \mapsto$ (all first, all last-first)

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
(0	44)	(0	44)	(31	31)	(56	56)	Step 1
(0	0)	(31	56)	(44	44)	(0	0)	Step 2
0	31	44	0	0	25	0	0	mob(α)
	31	44	0	0	12.5	0	0	$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$	31		0		12.5		0	Sh_1
$\sum$		44	0			0	0	Sh_2
$\sum$								Sh_3

Illustrating Fast Möbius/Yates Transformation

(first,last) $\cdots \mapsto$ (all first, all last-first)

$\emptyset$	1	2	1,2	3	1,3	2,3	1,2,3	α
(0	31)	(44	75)	(0	56)	(44	100)	val α
(0	44)	(0	44)	(31	31)	(56	56)	Step 1
(0	0)	(31	56)	(44	44)	(0	0)	Step 2
0	31	44	0	0	25	0	0	mob(α)
	31	44	0	0	12.5	0	0	$\lvert\alpha\rvert^{-1}$ mob(α)
$\sum$	31		0		12.5		0	Sh_1
$\sum$		44	0			0	0	Sh_2
$\sum$				0	12.5	0	0	Sh_3

Thank You!

Questions, Comments

`mailto:e.plischke@hzdr.de`

Preprints, Scripts, Stuff

`https://artefakte.rz-housing.tu-clausthal.de/epl/`

GitLab Repository

`https://gitlab.gwdg.de/elmar.plischke/global-sensitivity-analysis-collection`

References (I)

Goda, T. (2021).
A simple algorithm for global sensitivity analysis with Shapley effects.
Reliability Engineering&System Safety, 213:107702.

Good, I. J. (1958).
The interaction algorithms and practical Fourier analysis.
Journal of the Royal Statistical Society, Series B, 20:361–372.
Addendum: 22:372-375, 1960.

Grabisch, M. (2006).
Capacities and games on lattices: A survey of results.
International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 14(4):371–392.

Owen, A. B. (2014).
Sobol' indices and Shapley values.
SIAM/ASA Journal on Uncertainty Quantification, 2(1):245–251.

Plischke, E., Rabitti, G., and Borgonovo, E. (2021).
Computing Shapley effects for sensitivity analysis.
SIAM/ASA Journal on Uncertainty Quantification, 9(4):1411–1437.

Yates, F. (1937).
The design and analysis of factorial experiments.
Technical Communication 35, Imperial Bureau of Soil Science, Harpenden.