UTA-splines and UTADIS-splines

Olivier Sobrie[1,2] - Nicolas Gillis[2] - Vincent Mousseau[1] - Marc Pirlot[2]

[1]École Centrale de Paris - Laboratoire de Génie Industriel
[2]University of Mons - Faculty of engineering

July 5, 2016

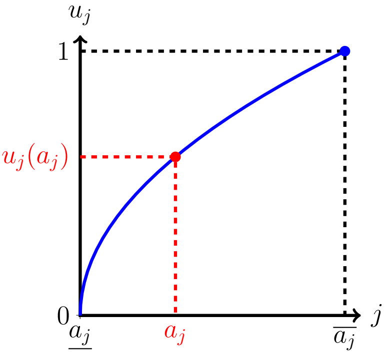
Additive value function model I

Principle

- A **score** is computed for each alternative
- The score is used to **rank** or to **sort** alternatives

Additive value function model II

- A **marginal value function** u_j is associated to each criterion j
- Marginal value functions are **monotonic**
- Marginal value functions are **normalized** between 0 and 1, s.t. $u_j(\underline{a_j}) = 0$ and $u_j(\overline{a_j}) = 1$
- A **weight** w_j is associated to each criterion j, s.t. $\sum_{j=1}^{n} w_j = 1$

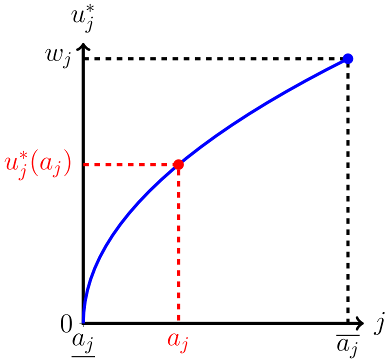


- Utility of an alternative a :

$$U(a) = \sum_{j=1}^{n} w_j \cdot u_j(a_j)$$

Additive value function model II

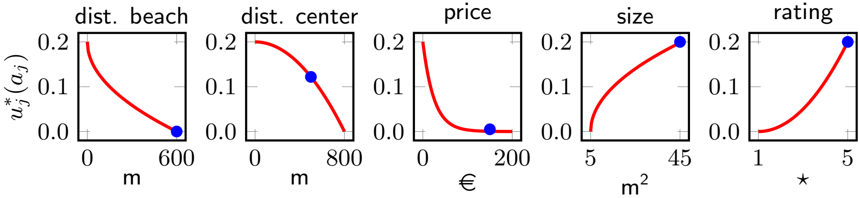
- A **marginal value function** u_j is associated to each criterion j
- Marginal value functions are **monotonic**
- Marginal value functions are **normalized** between 0 and 1, s.t. $u_j^*(\underline{a_j}) = 0$ and $u_j^*(\overline{a_j}) = w_j$
- A **weight** w_j is associated to each criterion j, s.t. $\sum_{j=1}^{n} w_j = 1$



- We also have : $u_j^*(a) = w_j \cdot u_j(a_j)$ and $u_j^*(\overline{a_j}) = w_j$
- Utility of an alternative a :

$$U(a) = \sum_{j=1}^{n} w_j \cdot u_j(a_j) = \sum_{j=1}^{n} u_j^*(a_j)$$

Example



Ranking

Plaza		Miramar		Hilton		Hotel W
	$\succ$		$\succ$		$\succ$	
0.53		0.51		0.43		0.41

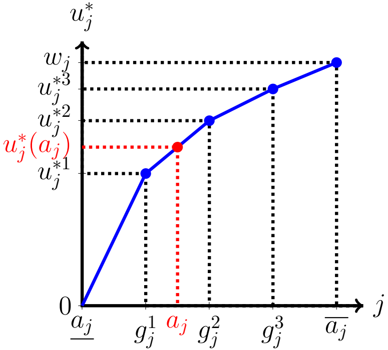
Sorting

Good 0.5 **Bad**

Plaza	Miramar	Hilton	Hotel W
			
0.53	0.51	0.43	0.41

Existing methods for learning an AVF model

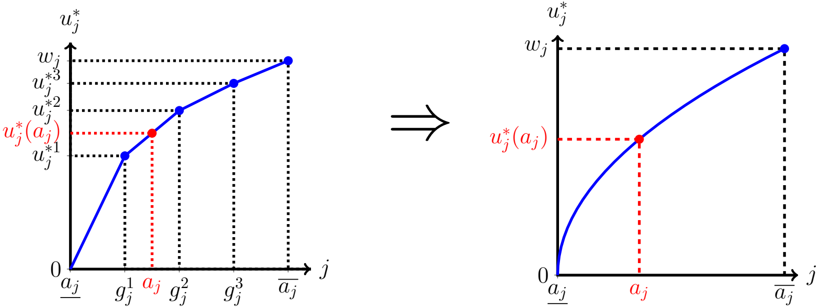
- **UTA** : LP for learning the parameters of an AVF-ranking model
- **UTADIS** : LP for learning the parameters of an AVF-sorting model
- Other methods : **UTA***, **ACUTA**, . . .
- **Monotonicity** of the marginals is ensured
- Marginals are modeled with **piecewise linear functions**



UTA(DIS)-poly I

Principle

- Use of **polynomials** for the marginal value functions



Motivations

- Improve the **flexibility** of the model
- Improve the **interpretability** of the model

UTA(DIS)-poly II

- Use of **semi-definite programming** (SDP)
  - Based on **interior point methods**
  - Possibility to impose the **nonnegativity of a symmetric matrix**

$$Q = \begin{pmatrix} q_{1,1} & q_{1,2} & q_{1,3} & \cdots & q_{1,n} \\ & q_{2,2} & q_{2,3} & \cdots & q_{2,n} \\ & & q_{3,3} & \cdots & q_{3,n} \\ & & & \ddots & \vdots \\ \text{(symmetric)} & & & & q_{n,n} \end{pmatrix} \geq 0$$

- Monotonicity of the marginals guaranteed
  - Ensured by imposing the **nonnegativity of the derivative**
  - Use of **Hilbert's theorems**

UTA(DIS)-poly III

Theorem (Hilbert)

A polynomial $F : \mathbb{R}^n \rightarrow \mathbb{R}$ is nonnegative if it is possible to decompose it as a sum of squares (SOS) :

$$F(z) = \sum_s f_s^2(z) \qquad \text{with } z \in \mathbb{R}^n.$$

Theorem (Hilbert)

A non-negative polynomial in one variable is always a SOS.

Theorem (Hilbert)

A polynomial $p(x)$ in one variable x is non-negative in the interval $[v_1, v_2]$, if and only if $p(x) = (x - v_1) \cdot q(x) + (v_2 - x) \cdot r(x)$ where $q(x)$ and $r(x)$ are SOS.

UTA-poly - Example I

	x	y
a^1	10	7
a^2	6	8
a^3	7	5

$$a^1 \succ a^2 \succ a^3$$

- We define $u_1^*(x)$ and $u_2^*(y)$ as **third degree polynomials** :

$$u_1^*(x) = p_{x,0} + p_{x,1} \cdot x + p_{x,2} \cdot x^2 + p_{x,3} \cdot x^3,$$
$$u_2^*(y) = p_{y,0} + p_{y,1} \cdot y + p_{y,2} \cdot y^2 + p_{y,3} \cdot y^3.$$

- **Scores** of a^1, a^2 and a^3 are given by :

$$U(a^1) = p_{x,0} + 10p_{x,1} + 100p_{x,2} + 1000p_{x,3} + p_{y,0} + 7p_{y,1} + 49p_{y,2} + 343p_{y,3},$$
$$U(a^2) = p_{x,0} + 6p_{x,1} + 36p_{x,2} + 324p_{x,3} + p_{y,0} + 8p_{y,1} + 64p_{y,2} + 512p_{y,3},$$
$$U(a^3) = p_{x,0} + 7p_{x,1} + 49p_{x,2} + 343p_{x,3} + p_{y,0} + 5p_{y,1} + 25p_{y,2} + 125p_{y,3}.$$

UTA-poly - Example II

- **Scores** of a^1, a^2 and a^3 are given by :

$$U(a^1) = p_{x,0} + 10p_{x,1} + 100p_{x,2} + 1000p_{x,3} + p_{y,0} + 7p_{y,1} + 49p_{y,2} + 343p_{y,3},$$
$$U(a^2) = p_{x,0} + 6p_{x,1} + 36p_{x,2} + 324p_{x,3} + p_{y,0} + 8p_{y,1} + 64p_{y,2} + 512p_{y,3},$$
$$U(a^3) = p_{x,0} + 7p_{x,1} + 49p_{x,2} + 343p_{x,3} + p_{y,0} + 5p_{y,1} + 25p_{y,2} + 125p_{y,3}.$$

- We have $a^1 \succ a^2$ **and** $a^2 \succ a^3$, which implies :

$$\begin{cases} U(a^1) - U(a^2) + \sigma^+(a^1) - \sigma^-(a^1) - \sigma^+(a^2) + \sigma^-(a^2) & > & 0, \\ U(a^2) - U(a^3) + \sigma^+(a^2) - \sigma^-(a^2) - \sigma^+(a^1) + \sigma^-(a^1) & > & 0. \end{cases}$$

- By replacing $U(a^1)$, $U(a^2)$ and $U(a^3)$, we have :

$$\begin{cases} 4p_{x,1} + 64p_{x,2} + 776p_{x,3} - p_{y,1} - 15p_{y,2} - 231p_{y,3} + \sigma^+(a^1) - \sigma^-(a^1) & & \\ \qquad -\sigma^+(a^2) + \sigma^-(a^2) & > & 0, \\ -p_{x,1} - 13p_{x,2} - 19p_{x,3} + 3p_{y,1} + 39p_{y,2} + 387p_{y,3} + \sigma^+(a^2) - \sigma^-(a^2) & & \\ \qquad -\sigma^+(a^3) + \sigma^-(a^3) & > & 0. \end{cases}$$

UTA-poly - Example III

- We impose the derivative of u_1^* and u_2^* to be **SOS** :

$$
\begin{aligned}
u_1^{*'} &= \overline{x}^{\mathsf{T}} Q \overline{x} \\
&= \begin{pmatrix} 1 \\ x \end{pmatrix}^{\mathsf{T}} \begin{pmatrix} q_{0,0} & q_{0,1} \\ q_{1,0} & q_{1,1} \end{pmatrix} \begin{pmatrix} 1 \\ x \end{pmatrix} \\
&= q_{0,0} + (q_{0,1} + q_{0,1})\, x + q_{1,1} x^2, \\
u_2^{*'} &= \overline{y}^{\mathsf{T}} R \overline{y} \\
&= r_{0,0} + (r_{0,1} + r_{1,0})\, y + r_{1,1} y^2.
\end{aligned}
$$

- Q and R have to be **semi-definite positive**, in conjunction with :

$$
\begin{cases} p_{x,1} &= q_{0,0}, \\ 2p_{x,2} &= q_{0,1} + q_{1,0}, \\ 3p_{x,3} &= q_{1,1}, \end{cases} \quad \text{and} \quad \begin{cases} p_{y,1} &= r_{0,0}, \\ 2p_{y,2} &= r_{0,1} + r_{1,0}, \\ 3p_{y,3} &= r_{1,1}. \end{cases}
$$

UTA-poly - Example IV

- We add **normalization** constraints :

$$\begin{cases} p_{x,0} &= 0, \\ p_{y,0} &= 0, \\ 10p_{x,1} + 100p_{x,2} + 1000p_{x,3} + 10p_{y,1} + 100p_{y,2} + 1000p_{y,3} &= 1. \end{cases}$$

UTA-poly - Example V

$$\min \sigma^+(a^1) + \sigma^-(a^1) + \sigma^+(a^2) + \sigma^-(a^2) + \sigma^+(a^3) + \sigma^-(a^3).$$

such that :

$$\left\{\begin{array}{rcl}
4p_{x,1} + 64p_{x,2} + 776p_{x,3} - p_{y,1} - 15p_{y,2} - 231p_{y,3} & & \\
+\sigma^+(a^1) - \sigma^-(a^1) - \sigma^+(a^2) + \sigma^-(a^2) & > & 0, \\
-p_{x,1} - 13p_{x,2} - 19p_{x,3} + 3p_{y,1} + 39p_{y,2} + 387p_{y,3} & & \\
+\sigma^+(a^2) - \sigma^-(a^2) - \sigma^+(a^3) + \sigma^-(a^3) & > & 0, \\
p_{x,0} & = & 0, \\
p_{y,0} & = & 0, \\
10p_{x,1} + 100p_{x,2} + 1000p_{x,3} + 10p_{y,1} + 100p_{y,2} + 1000p_{y,3} & = & 1, \\
p_{x,1} & = & q_{0,0}, \\
2p_{x,2} & = & q_{0,1} + q_{1,0}, \\
3p_{x,3} & = & q_{1,1}, \\
p_{y,1} & = & r_{0,0}, \\
2p_{y,2} & = & r_{0,1} + r_{1,0}, \\
3p_{y,3} & = & r_{1,1},
\end{array}\right.$$

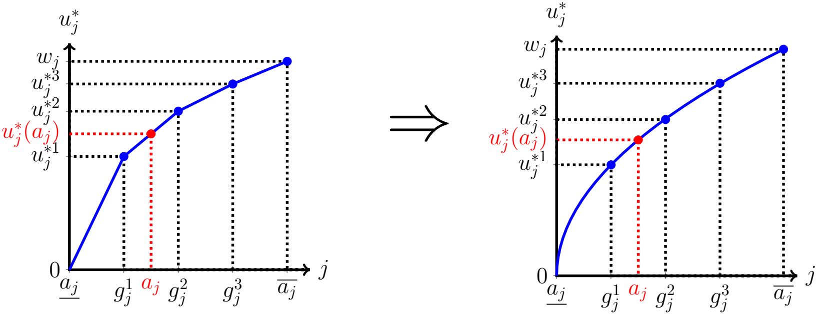
with :

$$\left\{\begin{array}{rcl}
Q, R & & PSD, \\
\sigma^+(a^1), \sigma^-(a^1), \sigma^+(a^2), \sigma^-(a^2), \sigma^+(a^3), \sigma^-(a^3) & \geq & 0.
\end{array}\right.$$

UTA(DIS)-splines I

Principle

- Use of **splines** for the marginal value functions
- **Monotonicity** of the marginals guaranteed as in UTA(DIS)-poly



Motivations

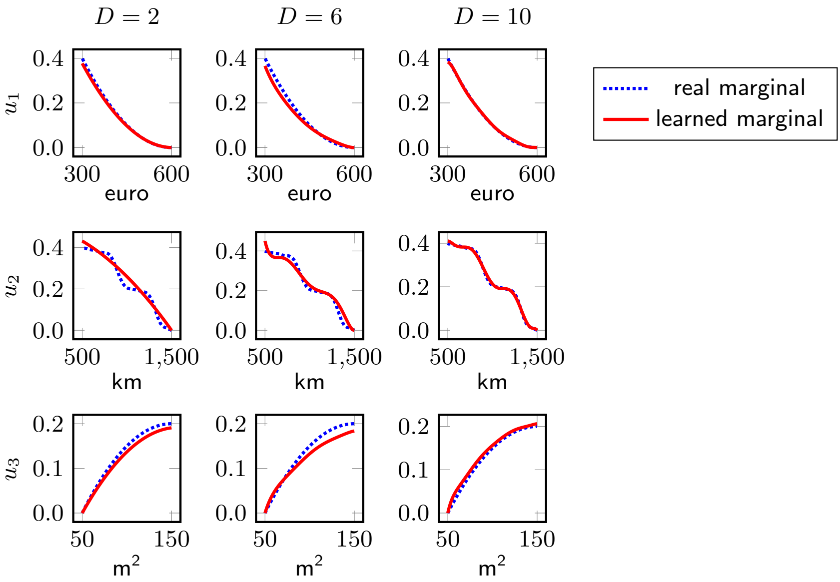
- Improve the **flexibility** of the model
- **Generalization** of UTA(DIS)-poly and UTA(DIS)

Implementation

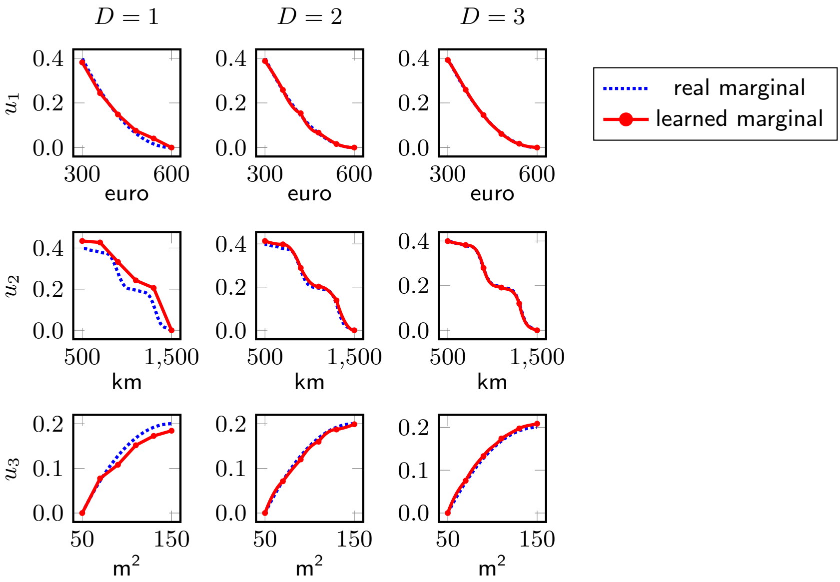
- Implemented in **MATLAB**
- **CVX** library
- **SDPT3** solver

- Everything is available on `https://github.com/oso/uta-matlab`

Example of marginals learned with UTA-poly



Example of marginals learned with UTA-splines



Test with real data sets

- 5 datasets containing **209 to 1728 instances**
- Instances evaluated on **4 to 10 criteria**
- Sorting of the alternatives in **4 to 5 categories**

Dataset	# instances	# attributes	# categories
JRA	172	5	4
CPU	209	6	4
LEV	1000	4	5
SWD	1000	10	4
CEV	1728	6	4

- Dataset split 100 times in two disjoint sets (**learning set** and **test set**)

Test with real data sets I

Test set (number of pieces - degree - continuity)

Size	Dataset	1-1-0	1-2-0	1-3-0
30 %	CEV	0.7449 ± 0.0110	0.7662 ± 0.0099	0.7487 ± 0.0109
	CPU	0.9195 ± 0.0350	0.9119 ± 0.0319	0.9050 ± 0.0315
	JRA	0.5950 ± 0.0537	0.6084 ± 0.0540	0.6248 ± 0.0399
	LEV	0.5935 ± 0.0134	0.5902 ± 0.0146	0.5768 ± 0.0164
	SWD	0.5408 ± 0.0174	0.5439 ± 0.0201	0.5395 ± 0.0186
50 %	CEV	0.7463 ± 0.0115	0.7668 ± 0.0130	0.7502 ± 0.0133
	CPU	0.9365 ± 0.0261	0.9275 ± 0.0272	0.9166 ± 0.0279
	JRA	0.6134 ± 0.0467	0.6177 ± 0.0487	0.6342 ± 0.0376
	LEV	0.5961 ± 0.0145	0.5933 ± 0.0159	0.5810 ± 0.0169
	SWD	0.5481 ± 0.0164	0.5495 ± 0.0186	0.5436 ± 0.0195
70 %	CEV	0.7444 ± 0.0141	0.7657 ± 0.0136	0.7473 ± 0.0139
	CPU	0.9408 ± 0.0235	0.9308 ± 0.0242	0.9234 ± 0.0285
	JRA	0.6309 ± 0.0476	0.6315 ± 0.0502	0.6405 ± 0.0414
	LEV	0.5982 ± 0.0189	0.5938 ± 0.0188	0.5859 ± 0.0197
	SWD	0.5479 ± 0.0183	0.5506 ± 0.0196	0.5470 ± 0.0162

Test with real data sets II

Test set (number of pieces - degree - continuity)

Size	Dataset	1-3-2	2-3-2	3-3-2
30 %	CEV	0.7487 ± 0.0109	0.7583 ± 0.0100	0.7603 ± 0.0115
	CPU	0.9050 ± 0.0315	0.9043 ± 0.0285	0.8988 ± 0.0296
	JRA	0.6248 ± 0.0399	0.6236 ± 0.0428	0.6197 ± 0.0478
	LEV	0.5768 ± 0.0164	0.5745 ± 0.0184	0.5738 ± 0.0180
	SWD	0.5395 ± 0.0186	0.5380 ± 0.0186	0.5380 ± 0.0191
50 %	CEV	0.7502 ± 0.0133	0.7607 ± 0.0096	0.7626 ± 0.0105
	CPU	0.9166 ± 0.0279	0.9179 ± 0.0290	0.9116 ± 0.0305
	JRA	0.6342 ± 0.0376	0.6348 ± 0.0410	0.6385 ± 0.0461
	LEV	0.5810 ± 0.0169	0.5780 ± 0.0161	0.5780 ± 0.0166
	SWD	0.5436 ± 0.0195	0.5433 ± 0.0184	0.5424 ± 0.0187
70 %	CEV	0.7473 ± 0.0139	0.7579 ± 0.0122	0.7597 ± 0.0139
	CPU	0.9234 ± 0.0285	0.9262 ± 0.0272	0.9208 ± 0.0273
	JRA	0.6405 ± 0.0414	0.6449 ± 0.0429	0.6482 ± 0.0451
	LEV	0.5859 ± 0.0197	0.5829 ± 0.0189	0.5815 ± 0.0178
	SWD	0.5470 ± 0.0162	0.5451 ± 0.0178	0.5445 ± 0.0171

Conclusion and further research

- **Computing time** in the same order of magnitude as UTA(DIS)
- **Generalization** of UTA(DIS)
- **Overfitting** effect

- Test with **other datasets**
- Use **semi-definite programming** with other MCDA methods

- Source code on `https://github.com/oso/uta-matlab`

Thank you for your attention !

References I