UTA-poly and UTADIS-poly: using polynomial marginal utility functions in UTA and UTADIS

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 15, 2015
1 UTA methods

2 Motivations for UTA-poly and UTADIS-poly

3 UTA-poly and UTADIS-poly

4 Experiments

5 Conclusion
Additive utility function model

- A marginal utility function $u_j$ is associated to each criterion $j$
- Marginal utility functions are **monotonic**
- Marginal utility functions are **normalized** between 0 and 1, s.t. $u_j(g_j) = 0$ and $u_j(\overline{g}_j) = 1$
- A **weight** $w_j$ is associated to each criterion $j$, s.t. $\sum_j w_j = 1$

Utility of an alternative $a$: $U(a) = \sum_{j=1}^{n} w_j \cdot u_j(a_j)$
Additive utility function model

- A marginal utility function $u_j$ is associated to each criterion $j$.
- Marginal utility functions are monotonic.
- Marginal utility functions are normalized between 0 and 1, s.t. $u_j(g_j) = 0$ and $u_j(\bar{g}_j) = 1$.
- A weight $w_j$ is associated to each criterion $j$, s.t. $\sum_j w_j = 1$.

We also have:
\[ u_j^*(a_j) = w_j \cdot u_j(a_j) \quad \text{and} \quad u_j^*(\bar{g}_j) = w_j \]

Utility of an alternative $a$:
\[
U(a) = \sum_{j=1}^n w_j \cdot u_j(a_j) = \sum_{j=1}^n u_j^*(a_j)
\]
UTA : Presentation

- Not easy to elicit directly the marginal utility functions
- **Disaggregation** procedure proposed by [Jacquet-Lagrèze and Siskos, 1982]
- Utility functions are computed on basis of a ranking given in input
- Linear programming
UTA - Constraints I

- Two type of information (pairwise comparison):
  1. \( a \) is preferred to \( b \), i.e. \( U(a) > U(b) \Rightarrow (a, b) \in \mathcal{P} \)
  2. \( a \) is indifferent to \( b \), i.e. \( U(a) = U(b) \Rightarrow (a, b) \in \mathcal{I} \)

- A potential error is introduced for each alternative utility \( U(a) \), s.t.
  \( U'(a) = U(a) + \sigma^+(a) - \sigma^-(a) \)

- Constraints of the linear program:

\[
\begin{align*}
U(a) - U(b) + \sigma^+(a) - \sigma^-(a) \\
-\sigma^+(b) + \sigma^-(b) &> 0 & \forall (a, b) \in \mathcal{P}, \\
U(a) - U(b) + \sigma^+(a) - \sigma^-(a) \\
-\sigma^+(b) + \sigma^-(b) &= 0 & \forall (a, b) \in \mathcal{I}, \\
\sum_{j=1}^{n} u_j^*(g_j) &= 1, \\
\sum_{j=1}^{n} u_j^*(g_j) &= 0, \\
\sigma^+(a), \sigma^-(a) &\geq 0 & \forall a \in A^*, \\
u_j^* &\text{ monotonic} & \forall j \in N.
\end{align*}
\]
Monotonicity is ensured by using **piecewise linear functions** for the marginal utility functions.

![Diagram showing piecewise linear functions and marginal utility values for alternatives](image)

- Domain of the criterion split in $k$ equal parts
- Position of the $g^l_j$, for $l = 0, \ldots, k$ fixed a priori (equidistant)

Marginal utility value of an alternative $a$:

$$u^*_j(a) = u^{*L-1}_j + \left(\frac{a_j - g^{L-1}_j}{g^L_j - g^{L-1}_j}\right) \left(u^{*L}_j - u^{*L-1}_j\right)$$

with $g^L_j$ the first breakpoint s.t. $a_j \leq g^L_j$
UTADIS

- Sorting problems
- Comparison of alternatives to thresholds delimiting the categories
- Disaggregation though linear programming
- Similar approach as for UTA: utility functions are modelled through piecewise linear functions
Motivations for UTA-poly and UTADIS-poly I

Drawbacks of UTA methods

- Marginal utility functions are not natural close to the breakpoints of the piecewise linear functions
- Breakpoints at pre-defined position: limit the flexibility of the model

Existing works


Motivations for UTA-poly and UTADIS-poly

We propose to replace piecewise linear functions by polynomial ones by using semi-definite programming (SDP)

Degree of the polynomial chosen a priori

\[ u^*_j(a_j) = u^*_j(g^1_j) + \left( \frac{a_j - g^1_j}{g^2_j - g^1_j} \right) (u^*_2 - u^*_1) \]

\[ u^*_j(a) = p_0 + p_1 \cdot a_j + p_2 \cdot a_j^2 + \ldots + p_D \cdot a_j^D \]
A polynomial \( F(z) \), with \( z \in \mathbb{R}^n \) is nonnegative if it is possible to decompose it as a sum of squares (SOS) :

\[
F(z) = \sum_s f_s^2(z) \quad \text{with} \quad f_s(z) \in \mathbb{R}^n.
\]

Not every non-negative polynomial is a Sum Of Squares (SOS), but :

**Theorem** *(Hilbert)* A non-negative polynomial in one variable is always a SOS.
Consider the following polynomial of degree $D$:

$$p(x) = p_0 + p_1 x + p_2 x^2 + \ldots + p_D x^D = \sum_{i=0}^{D} p_i x^i.$$  

$p(x)$ non-negative $\iff$ it can be decomposed as a SOS.

Let $d = \left\lceil \frac{D}{2} \right\rceil$, $b_s^T = [b_s^0, b_s^1, \ldots, b_s^d]$ and $\bar{x}^T = [1, x, \ldots, x^d]$, the polynomial reads:

$$p(x) = \sum_s q_s^2(x) = \sum_s \left[ \sum_{i=0}^{d} b_s^i x^i \right] = \sum_s \left( b_s^T \bar{x} \right)^2$$

$$= \sum_s \bar{x}^T b_s b_s^T \bar{x} = \bar{x}^T \left[ \sum_s b_s b_s^T \right] \bar{x} = \bar{x}^T Q \bar{x}$$
Semi-Definite Programming (SDP) III

\[ p(x) = p_0 + p_1 x + p_2 x^2 + \ldots + p_D x^D \]

\[
= \bar{x}^T Q \bar{x} = \begin{pmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^d \end{pmatrix}^T \begin{pmatrix} q_{0,0} & q_{0,1} & q_{0,2} & \ldots & q_{0,d} \\ q_{1,0} & q_{1,1} & q_{1,2} & \ldots & q_{1,d} \\ q_{2,0} & q_{2,1} & q_{2,2} & \ldots & q_{2,d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ q_{d,0} & q_{d,1} & q_{d,2} & \ldots & q_{d,d} \end{pmatrix} \begin{pmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^d \end{pmatrix}.
\]

\[ p_0, p_1, \ldots, p_D \] are obtained by summing the off-diagonal entries of \( Q \) :

\[
\begin{cases}
p_0 = q_{0,0}, \\
p_1 = q_{1,0} + q_{0,1}, \\
p_2 = q_{2,0} + q_{1,1} + q_{0,2}, \\
\vdots \\
p_{2d-1} = q_{d,d-1} + q_{d-1,d}, \\
p_{2d} = q_{d,d}, \text{if } D \text{ is odd, we have } p_{2d} = 0.
\end{cases}
\]
Marginal utility functions defined as polynomials of degree $D$:

$$u_j^*(a_j) = \sum_{i=0}^{D} p_{j,i} \cdot a_j^i.$$  

To ensure monotonicity of the function, $u_j^{*'}(a_j)$ has to be nonnegative.  
\Rightarrow $u_j^{*'}$ should be a SOS:

$$\frac{\partial u_j^*}{\partial a_j} = p_{j,1} + 2p_{j,2} \cdot a_j + 3p_{j,3} \cdot a_j^2 + \ldots + Dp_{j,n} \cdot a_j^{D-1}$$

$$= \frac{\partial}{\partial a_j} \left[ \bar{a}_j^\top Q \bar{a}_j \right]$$

Using a SDP solver, we impose $Q$ to be semidefinite positive.
We define $u_1^*(x)$ and $u_2^*(y)$ as second degree polynomials:

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

Utilities of $a^1$, $a^2$ and $a^3$ are given by:

$$U(a^1) = p_{x,0} + 10p_{x,1} + 100p_{x,2} + p_{y,0} + 7p_{y,1} + 49p_{y,2},$$
$$U(a^2) = p_{x,0} + 6p_{x,1} + 36p_{x,2} + p_{y,0} + 8p_{y,1} + 64p_{y,2},$$
$$U(a^3) = p_{x,0} + 7p_{x,1} + 49p_{x,2} + p_{y,0} + 5p_{y,1} + 25p_{y,2}.$$
UTA-poly - Example II

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

$$\left\{ \begin{array}{c} U(a^1) - U(a^2) + \sigma^+(a^1) - \sigma^-(a^1) - \sigma^+(a^2) + \sigma^-(a^2) > 0, \\ U(a^2) - U(a^3) + \sigma^+(a^2) - \sigma^-(a^2) - \sigma^+(a^1) + \sigma^-(a^1) > 0. \end{array} \right.$$ 

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

$$\left\{ \begin{array}{c} 4p_{x,1} + 64p_{x,2} - p_{y,1} - 15p_{y,2} + \sigma^+(a^1) - \sigma^-(a^1) - \sigma^+(a^2) + \sigma^-(a^2) > 0, \\ -p_{x,1} - 13p_{x,2} + 3p_{y,1} + 39p_{y,2} + \sigma^+(a^2) - \sigma^-(a^2) - \sigma^+(a^3) + \sigma^-(a^3) > 0. \end{array} \right.$$
We impose the derivate of $u_1^*$ and $u_2^*$ to be SOS:

$$\frac{\partial u_1^*}{\partial x} = \bar{x}^T Q \bar{x}$$

$$= \begin{pmatrix} 1 \\ \bar{x} \end{pmatrix}^T \begin{pmatrix} q_{x,0,0} & q_{x,0,1} \\ q_{x,1,0} & q_{x,1,1} \end{pmatrix} \begin{pmatrix} 1 \\ \bar{x} \end{pmatrix}$$

$$= q_{0,0} + (q_{0,1} + q_{0,1}) \bar{x} + q_{1,1} \bar{x}^2,$$

$$\frac{\partial u_2^*}{\partial y} = \bar{y}^T R \bar{y}$$

$$= r_{0,0} + (r_{0,1} + r_{1,0}) y + r_{1,1} y^2.$$

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

$$\begin{cases}
  p_{x,1} = q_{0,1} + q_{1,0}, \\
  2p_{x,2} = q_{1,1},
\end{cases}$$

and

$$\begin{cases}
  p_{y,1} = r_{0,1} + r_{1,0}, \\
  2p_{y,2} = r_{1,1}.
\end{cases}$$
We add normalization constraints:

\[
\begin{align*}
\rho_{x,0} &= 0, \\
\rho_{y,0} &= 0, \\
10\rho_{x,1} + 100\rho_{x,2} + 10\rho_{y,1} + 100\rho_{y,2} &= 1.
\end{align*}
\]
Finally, we obtain the following program:

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

such that:

$$\begin{cases} 
4p_{x,1} + 64p_{x,2} - p_{y,1} - 15p_{y,2} + \sigma^+(a^1) - \sigma^-(a^1) \\
- \sigma^+(a^2) + \sigma^-(a^2) & > 0, \\
- p_{x,1} - 13p_{x,2} + 3p_{y,1} + 39p_{y,2} + \sigma^+(a^2) - \sigma^-(a^2) \\
- \sigma^+(a^3) + \sigma^-(a^3) & > 0, \\
p_{x,0} & = 0, \\
p_{y,0} & = 0, \\
10p_{x,1} + 100p_{x,2} + 10p_{y,1} + 100p_{y,2} & = 1, \\
p_{x,1} & = q_{0,1} + q_{1,0}, \\
2p_{x,2} & = q_{1,1}, \\
p_{y,1} & = r_{0,1} + r_{1,0}, \\
2p_{y,2} & = r_{1,1},
\end{cases}$$

with:

$$\begin{cases} 
Q, R & \geq 0, \\
\sigma^+(a^1), \sigma^-(a^1), \sigma^+(a^2), \sigma^-(a^2), \sigma^+(a^3), \sigma^-(a^3), & \geq 0.
\end{cases}$$
Experiments

Illustrative example (UTADIS-poly) I

- Evaluation of accommodations for holidays on 3 criteria: price, distance, and size.
- 10 categories (from good to bad)
- Consider the following true marginal utility functions

![Price Utility](image1.png)

- 200 examples given as input to UTADIS-poly
Illustrative example (UTADIS-poly) II

Experiments

University of Mons

O. Sobrie - N. Gillis - V. Mousseau - M. Pirlot - July 15, 2015 20 / 26
Computing time

(a) UTA-polynomial

(b) UTA-polynomial

(c) UTA-polynomial

(d) UTA
Learning a UTA-poly model from a ranking obtained with UTA

(a) Spearman distance

- $n = 5; D = 3$
- $n = 5; D = 7$
- $n = 10; D = 3$

(b) Kendall Tau

- $m = 100; D = 3$
- $m = 100; D = 7$
- $m = 200; D = 3$

(c) degree of polynomials

- $m = 100; n = 5$
- $m = 100; n = 10$
- $m = 200; n = 5$
Learning a UTADIS-poly model from assignment examples obtained with UTADIS

Experiments

(a) Learning a UTADIS-poly model from assignment examples obtained with UTADIS.

(b) Classification accuracy for different numbers of criteria.

(c) Classification accuracy for different numbers of categories.

(d) Classification accuracy for different degrees of the polynomials.

University of Mons

Conclusion

- More natural marginal utility functions
- Do not increase computing time
- Some marginal utility functions are difficult to model (step)
Thank you for your attention!
