On belief functions implementations

Arnaud Martin

Arnaud.Martin@univ-rennes1.fr

Université de Rennes 1 - IRISA, Lannion, France

Xi'an, July, 9th 2017

INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTEMES ALÉATOIRES

Plan

- Natural order
- Smets codes
- General framework
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - probabilistic based model

Plan

- Natural order
- Smets codes
- General framework
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - probabilistic based model

Natural order or Binary order

Discernment frame: $\Omega = \{\omega_1, \omega_2, \dots, \omega_n\}$
Power set: all the disjunctions of Ω:
$2^\Omega = \{\emptyset, \{\omega_1\}, \{\omega_2\}, \{\omega_1 \cup \omega_2\}, \dots, \Omega\}$

Natural order:

$$
\begin{aligned}
2^\Omega \;=\;& \{\emptyset, \{\omega_1\}, \{\omega_2\}, \{\omega_1 \cup \omega_2\},\\
&\{\omega_3\}, \{\omega_1 \cup \omega_3\}, \{\omega_2 \cup \omega_3\}, \{\omega_1 \cup \omega_2 \cup \omega_3\},\\
&\{\omega_4\}, \dots, \Omega\}
\end{aligned}
$$

Natural order or Binary order

Natural order:

$\emptyset$ 0	ω_1 1	ω_2 2	$\omega_1 \cup \omega_2$ $3 = 2^2 - 1$
ω_3 $4 = 2^{3-1}$	$\omega_1 \cup \omega_3$ 5	$\omega_2 \cup \omega_3$ 6	$\omega_1 \cup \omega_2 \cup \omega_3$ $7 = 2^3 - 1$
ω_4 $8 = 2^{4-1}$	$\ldots$	$\ldots$	$\ldots$
ω_i 2^{i-1}	$\ldots$	$\ldots$	Ω $2^n - 1$

Bba in Matlab:

Example: $m_1(\omega_1) = 0.5$, $m_1(\omega_3) = 0.4$, $m_1(\omega_1 \cup \omega_2 \cup \omega_3) = 0.1$
$m_2(\omega_3) = 0.4$, $m_2(\omega_1 \cup \omega_3) = 0.6$

```
F1=[1 4 7]';
F2=[4 5]';
M1=[0.5 0.4 0.1]';
M2=[0.4 0.6]';
```

Combination

$$m_{\text{Conj}}(X) = \sum_{Y_1 \cap Y_2 = X} m_1(Y_1) m_2(Y_2) \quad (1)$$

$\omega_1 \cap (\omega_1 \cup \omega_3)$: $1 \cap 5$
In binary with 3 digits for a frame of 3 elements: 1=001 and 5=101=001 |011
$001\&101 = 001$

Natural order or Binary order

In Matlab:

```
sizeDS=3;
F1=[1 4 7]';
F2=[4 5]';
M1=[0.5 0.4 0.1]';
M2=[0.4 0.6]';
Fres=[];
Mres=[];
for i=1:size(F1)
   for j=1:size(F2)
      Fres=[Fres bi2de(de2bi(F1(i),sizeDS)&de2bi(F2(j),sizeDS))];
      Mres=[Mres M1(i)*M2(j)];
   end
end
```

Plan

- Natural order
- Smets codes
- General framework
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - probabilistic based model

Plan

- Natural order
- Smets codes
- General framework
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - probabilistic based model

Smets codes for DST

Smets gaves the codes of the Mobius transform (see Only_Mobius_Transf) for conversions:

- bba and belief: mtobel, beltom
- bba and plausibility: mtopl, pltom
- bba and communality: mtoq, qtom
- bba and implicability: mtob, btom
- bba to pignistic probability: mtobetp
- etc...

e.g. in Matlab:
m1=[0 0.4 0.1 0.2 0.2 0 0 0.1]';
mtobel(m1)
gives: 0 0.4000 0.1000 0.7000 0.2000 0.6000 0.3000 1.0000

Smets code

For s bbas m_j

Conjunctive combination

$$m_{\text{Conj}}(X) = \sum_{Y_1 \cap \ldots \cap Y_s = X} \prod_{j=1}^{s} m_j(Y_j)$$

The practical way:

$$q(X) = \prod_{j=1}^{s} q_j(X)$$

Disjunctive combination

$$m_{\text{Dis}}(X) = \sum_{Y_1 \cup \ldots \cup Y_s = X} \prod_{j=1}^{s} m_j(Y_j)$$

The practical way:

$$b(X) = \prod_{j=1}^{s} b_j(X)$$

DST code

In Matlab

For the conjunctive rule of combination:

```
m1=[0 0.4 0.1 0.2 0.2 0 0 0.1]';
m2=[0 0.2 0.3 0.1 0.1 0 0.2 0.1]';
q1=mtoq(m1);
q2=mtoq(m2);
qConj=q1.*q2;
mConj=qtom(qConj)
mConj =
   0.4100 0.2200 0.2000 0.0500 0.0900 0 0.0200 0.0100
```

DST code

In Matlab

For the disjunctive rule of combination:

```
m1=[0 0.4 0.1 0.2 0.2 0 0 0.1]';
m2=[0 0.2 0.3 0.1 0.1 0 0.2 0.1]';
b1=mtob(m1);
b2=mtob(m2);
bDis=b1.*b2;
mDis=btom(bDis)
mDis =
   0 0.0800 0.0300 0.3100 0.0200 0.0800 0.1300 0.3500
```

DST code

Once bbas are combined, to decide just use the functions mtobel, mtopl or mtobetp, etc.

In Matlab

```
mtopl(mConj)
  0 0.2800 0.2800 0.5000 0.1200 0.3900 0.3700 0.5900
mtobetp(mConj)
  0.4209 0.4040 0.1751

mtopl(mDis)
  0 0.8200 0.8200 0.9800 0.5800 0.9700 0.9200 1.0000
mtobetp(mDis)
  0.3917 0.3667 0.2417
```

DST code

DST code for the combination:

- criteria=1 Smets criteria
- criteria=2 Dempster-Shafer criteria (normalized)
- criteria=3 Yager criteria
- criteria=4 disjunctive combination criteria
- criteria=5 Dubois criteria (normalized and disjunctive combination)
- criteria=6 Dubois and Prade criteria (mixt combination)
- criteria=7 Florea criteria
- criteria=8 PCR6
- criteria=9 Cautious Denoeux Min for non-dogmatics functions
- criteria=10 Cautious Denoeux Max for separable functions
- criteria=11 Hard Denoeux for sub-normal functions
- criteria=12 Mean of the bbas

DST code

decisionDST code for the decision:

- criteria=1 maximum of the plausibility
- criteria=2 maximum of the credibility
- criteria=3 maximum of the credibility with rejection
- criteria=4 maximum of the pignistic probability
- criteria=5 Appriou criteria

DST code

test.m:

```
m1=[0 0.4 0.1 0.2 0.2 0 0 0.1]';
m2=[0 0.2 0.3 0.1 0.1 0 0.2 0.1]';
m3=[0.1 0.2 0 0.4 0.1 0.1 0 0.1]';

m3d=discounting(m3,0.95);

M_comb_Smets=DST([m1 m2 m3d],1);
M_comb_PCR6=DST([m1 m2],8);

class_fusion=decisionDST(M_comb_Smets',1)
class_fusion=decisionDST(M_comb_PCR6',1)
class_fusion=decisionDST(M_comb_Smets',5,0.5)
```

Plan

- Natural order
- Smets codes
- General framework
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - probabilistic based model

Plan

- Natural order
- Smets codes
- General framework
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - probabilistic based model

General framework

- Main problem of the DST code: all elements must be coded (not only the focal elements)
- Only usable for belief functions defined on power set (2^Ω)
- General belief functions framework works for power set and hyper power set (D^Ω)

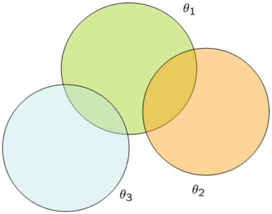
DSmT

DSmT introduced by Dezert, 2002.

- D^Ω closed set by union and intersection operators
- D^Ω is not closed by complementary, $A \in D^\Omega \not\Rightarrow \overline{A} \in D^\Omega$
- if $|\Omega| = n$: $2^\Omega << D^\Omega << 2^{2^\Omega}$
- D_r^Ω: reduced set considering some constraints $(\omega_2 \cap \omega_3 \equiv \emptyset)$

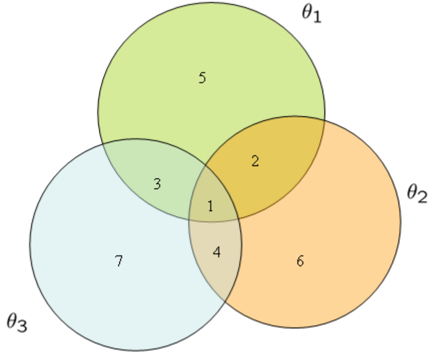
$$\mathrm{GPT}(X) = \sum_{Y \in D_r^\Omega, Y \not\equiv \emptyset} \frac{\mathcal{C}_\mathcal{M}(X \cap Y)}{\mathcal{C}_\mathcal{M}(Y)} m(Y)$$

where $\mathcal{C}_\mathcal{M}(X)$ is the cardinality of X in D_r^Ω



A simple codification (Martin, 2009)

Affect an integer of $[1; 2^n - 1]$ to each distinct part of Venn diagram ($n = |\Omega|$)

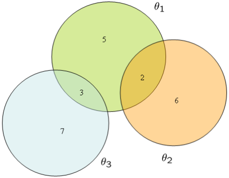


$$\Omega = \{[1\ 2\ 3\ 5], [1\ 2\ 4\ 6], [1\ 3\ 4\ 7]\}$$

General framework: codification

Adding a constraint: if $\Omega = \{[1\ 2\ 3\ 5], [1\ 2\ 4\ 6], [1\ 3\ 4\ 7]\}$ and we know $\omega_2 \cap \omega_3 \equiv \emptyset$ (*i.e.* $\omega_2 \cap \omega_3 \notin D_r^\Omega$)
The parts 1 and 4 of Venn diagram do not exist:

$$\Omega_r = \{[2\ 3\ 5], [2\ 6], [3\ 7]\}$$



Operations on focal elements

$$\omega_1 \cap \omega_3 = [3]$$
$$\omega_1 \cup \omega_3 = [2\ 3\ 5\ 7]$$
$$(\omega_1 \cap \omega_3) \cup \omega_2 = [2\ 3\ 6]$$

The cardinality $\mathcal{C}_\mathcal{M}(X)$: the number of intergers in the codification of X

Gives an easy Matlab programmation of the combination rules and the decision functions

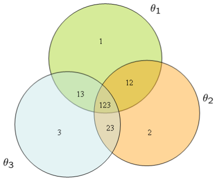
General framework: codification

Decoding: to present the decision or a result to the human - *The codification is not understandable*

If the decision set is given, we just have to sweep the corresponding part of D_r^Ω

Without any knowledge of the element to decode:

1. We can use the Smarandache condification more lisible but less practical in Matlab



2. We sweepp all D_r^Ω (first considering 2^Ω). There is a combinatorial risk.

General framework

Description of the problem frame={'A','B','C','D'};

```
% list of experts with focal elements and associated bba
  expert(1).name='Source 1';
  expert(1).focal={'1' '2u3u4' '1u2u3u4'};
  expert(1).bba=[0.47 0.18 0.35];
  expert(1).discount=1; % No discount

  expert(2).name='Source 2';
  expert(2).focal={'1' '2' '1u3' '1u2u3'};
  expert(2).bba=[0.3 0.4 0.2 0.1];
  expert(2).discount=0.1; % high discount

constraint={''}; % set of empty elements e.g. '1n2'
```

In test.m

Description of the problem

elemDec={'A'}; % set of decision elements:

- list of elements on which we can decide,
- A for all,
- S for singletons only,
- F for focal elements only,
- SF for singleton plus focal elements,
- Cm for given specificity, *e.g.* elemDec={'Cm' '1' '4'}; minimum of cardinality 1, maximum=4,
- 2T for only 2^Ω (DST case)

Parameters

Combination citerium

criteriumComb = is the combination criterium

- criteriumComb=1 Smets criterium
- criteriumComb=2 Dempster-Shafer criterium (normalized)
- criteriumComb=3 Yager criterium
- criteriumComb=4 disjunctive combination criterium
- criteriumComb=5 Florea criterium
- criteriumComb=6 PCR6
- criteriumComb=7 Mean of the bbas

Parameters

Combination citerium

criteriumComb = is the combination criterium

- criteriumComb=8 Dubois criterium (normalized and disjunctive combination)
- criteriumComb=9 Dubois and Prade criterium (mixt combination)
- criteriumComb=10 Mixt Combination (Martin and Osswald criterium)
- criteriumComb=11 DPCR (Martin and Osswald criterium)
- criteriumComb=12 MDPCR (Martin and Osswald criterium)
- criteriumComb=13 Zhang's rule

Parameters

Decision citerium

criteriumDec = is the combination criterium

- criteriumDec=0 maximum of the bba
- criteriumDec=1 maximum of the pignistic probability
- criteriumDec=2 maximum of the credibility
- criteriumDec=3 maximum of the credibility with reject
- criteriumDec=4 maximum of the plausibility
- criteriumDec=5 Appriou criterium
- criteriumDec=6 DSmP criterium

Parameters

Mode of fusion

mode='static'; % or 'dynamic'

Display

display = kind of display

- display = 0 for no display,
- display = 1 for combination display,
- display = 2 for decision display,
- display = 3 for both displays,
- display = 4 for a comparison decision display,
- display = 5 for a comparison decision display with figures

Backup

nameTest = name of the test. No parameter, no backup
Wframe = to display with the complet frame legend

General framework

Fusion

fuse(expert,constraint,frame,criteriumComb,criteriumDec,mode, elemDec,display)

Called functions:

- Coding: call coding, addConstraint, codingExpert
- Combination: call combination
- Decision: call decision
- Display: call decodingExpert, decodingFocal

Plan

- Natural order
- Smets codes
- General framework
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - Probabilistic based model

Plan

- Natural order
- Smets codes
- General framework
- How to obtain bbas?
    - Random bbas
    - Distance based model
    - Probabilistic based model

Random bbas (see RandomMass.m)

In Matlab:

1. OmegaSize=3;
2. nbFocalElement=4;
3. ind=randperm(2ˆOmegaSize);
4. indFocalElement=ind(1:nbFocalElement);
5. randMass=diff([0; sort(rand(nbFocalElement-1,1)); 1]);
   We take the difference between 3 ordered random numbers in [0,1], e.g. diff([0; [0.3; 0.9] ; 1]) gives 0.3 0.6 0.1
6. MasseOut(indFocalElement,i)=randMass;

Random bbas (see RandomMass.m)

- focal elements can be evrywhere:
  ind=randperm(2^OmegaSize);
- focal elements not on the emptyset:
  ind =1+randperm(2^OmegaSize-1);
- no dogmatic mass: one focal element is on Omega (ignorance):
  ind =[2^OmegaSize randperm(2^OmegaSize-1)];
- no dogmatic mass: one focal element is on Omega (ignorance) and focal elements are not on the emptyset
  ind =[2^OmegaSize (1+randperm(2^OmegaSize-2))]
- all the focal elements are the singletons:

```
ind=[ ];
for i=1:OmegaSize
   ind=[ind; 1+2^(i-1)];
end
```

Distance based model (Denœux 1995)

Only ω_i and Ω are focal elements, $n * m$ sources (experts)

- Prototypes case ($\mathbf{x}_i$ center of ω_i). For the observation x

$$\begin{aligned} m_j^i(\omega_i) &= \alpha_{ij} \exp[-\gamma_{ij} d^2(x, \mathbf{x}_i)] \\ m_j^i(\Omega) &= 1 - \alpha_{ij} \exp[-\gamma_{ij} d^2(x, \mathbf{x}_i)] \end{aligned}$$

  - $0 \leq \alpha_{ij} \leq 1$: discounting coefficient and $\gamma_{ij} > 0$, are parameters to play on the quantity of ignorance and on the form of the mass functions
  - The distance allows to give a mass to x higer according to the proximity to ω_i
- belief k-nn: we consider the k-nearest neigborhs insteed to $\mathbf{x}_i$
- Then we combine the bbas

Distance based model (Denœux 1995)

In Matlab (Denœux codes)

See ExampleIris.m

```
load iris
ind=randperm(150);
xapp=x(ind(1:100),:);
Sapp=S(ind(1:100));
xtst=x(ind(101:150),:);
Stst=S(ind(101:150));
[gamm,alpha] = knndsinit(xapp,Sapp); % initialization
[gamm,alpha,err] = knndsfit(xapp,Sapp,5,gamm,0); % parameter optimization
[m,L] =knndsval(xapp,Sapp,5,gamm,alpha,0,xtst); % test
[value,Sfind]=max(m);
[mat_conf,vect_prob_classif,vect_prob_error]=build_conf_matrix(Sfind,Stst)
```

Probabilistic based model $^{(1/6)}$

- Need to estimate $p(S_j|\omega_i)$
- 2 models proposed by Appriou according to both axioms:
  1. the $n * m$ couples $[M_i^j, \alpha_{ij}]$ are distinct information sources where focal elements are: ω_i, ω_i^c and Ω
  2. If $M_i^j = 0$ and the information is valid $(\alpha_{ij} = 1)$ then it is certain that ω_i is not true.

Probabilistic based model (2/6)

Model 1: $m_j^i(\omega_i) = M_i^j$
$m_j^i(\omega_i^c) = 1 - M_i^j$

Model 2: $m_j^i(\Omega) = M_i^j$
$m_j^i(\omega_i^c) = 1 - M_i^j$

Adding the reliability α_{ij} with the discounting:

Model 1:

$$
\begin{aligned}
m_j^i(\omega_i) &= \alpha_{ij} M_i^j \\
m_j^i(\omega_i^c) &= \alpha_{ij}(1 - M_i^j) \\
m_j^i(\Omega) &= 1 - \alpha_{ij}
\end{aligned}
$$

Model 2:

$$
\begin{aligned}
m_j^i(\omega_i) &= 0 \\
m_j^i(\omega_i^c) &= \alpha_{ij}(1 - M_i^j) \\
m_j^i(\Omega) &= 1 - \alpha_{ij}(1 - M_i^j)
\end{aligned}
$$

Probabilistic based model $^{(3/6)}$

How to find M_i^j?
3th axiom:

3 Conformity to the Bayesian approach (case where $p(S_j|\omega_j)$ is exactly the reality ($\alpha_{ij} = 1$) for all i, j) and all the *a priori* probabilities $p(\omega_i)$ are known)

Probabilistic based model $^{(4/6)}$

Model 1: $M_i^j = \frac{R_j p(S_j|\omega_j)}{1 + R_j p(S_j|\omega_j)}$

$$\begin{aligned} m_j^i(\omega_i) &= \frac{\alpha_{ij} R_j p(S_j|\omega_j)}{1 + R_j p(S_j|\omega_j)} \\ m_j^i(\omega_i^c) &= \frac{\alpha_{ij}}{1 + R_j p(S_j|\omega_j)} \\ m_j^i(\Omega) &= 1 - \alpha_{ij} \end{aligned}$$

with $R_j \geq 0$ a normalization factor.

Probabilistic based model $^{(5/6)}$

Model 2: $M_i^j = R_j p(S_j|\omega_j)$

$$
\begin{array}{rcl}
m_j^i(\omega_i) & = & 0 \\
m_j^i(\omega_i^c) & = & \alpha_{ij}(1 - R_j p(S_j|\omega_j)) \\
m_j^i(\Omega) & = & 1 - \alpha_{ij}(1 - R_j p(S_j|\omega_j)
\end{array}
$$

with $R_j \in [0, (\max_{S_j,i}(p(S_j|\omega_j)))^{-1}]$

In practical:

- α_{ij}: discounting coefficient fixed near 1 and $p(S_j|\omega_j)$ can be given by the confusion matrix
- Adapted to the cases where we learn one class against all the others

Probabilistic based model (6/6)

In Malab:

- take the previous confusion matrix or mat_conf=[68 12 22 ; 9 42 5 ; 8 2 87]
- mat_mass= bbaType(mat_conf,alpha,model): gives all the possible bbas (*i.e.* number of classes) for the given confusion matrix, alpha (a constant such as 0.95) and the model (1 or 2)
- bbas=buildBbas(Stst,mat_conf,alpha,model): gives the bbas resulting of the founded classes given in Stst

Probabilistic vs Distance

Difficulties:

- Appriou: learning the probabilities $p(S_j|\omega_j)$
- Denœux: choice of the distance $d(x, \mathbf{x}_i)$

Easiness:

- $p(S_j|\omega_j)$ easier to estimate on decisions with the confusion matrix of the classifiers
- $d(x, \mathbf{x}_i)$ easier to choose on the numeric outputs of classifiers (ex.: Euclidean distance)

References

- Toolboxes: on http://www.bfasociety.org
  iBelief: R package:
  https://cran.rstudio.com/web/packages/ibelief/index.html
- a lot of papers on: http://www.bfasociety.org

References

- On the presented codes:

  - Kennes R. and Smets Ph. (1991) Computational Aspects of the Möbius Transformation. Uncertainty in Artificial Intelligence 6, P.P. Bonissone, M. Henrion, L.N. Kanal, J.F. Lemmer (Editors), Elsevier Science Publishers (1991) 401-416.
  - A. Martin, Implementing general belief function framework with a practical codification for low complexity, in Advances and Applications of DSmT for Information Fusion, American Research Press Rehoboth, pp. 217-273, 2009.
  - T. Denœux. A k-nearest neighbor classification rule based on Dempster-Shafer theory. IEEE Transactions on Systems, Man and Cybernetics, 25(05):804-813, 1995.
  - L. M. Zouhal and T. Denœux. An evidence-theoretic k-NN rule with parameter optimization. IEEE Transactions on Systems, Man and Cybernetics - Part C, 28(2):263-271,1998.
  - Appriou, Discrimination multisignal par la théorie de l'évidence, chap 7, Décision et Reconnaissance des formes en signal, Hermes Science Publication, 2002, 219-258

References

- Other way to code belief functions:

  - P.P. Shenoy and G. Shafer. Propagating belief functions with local computations. IEEE Expert, 1(3):43-51, 1986.
  - R. Haenni and N. Lehmann. Implementing belief function computations. International Journal of Intelligent Systems, Special issue on the Dempster-Shafer theory of evidence, 18(1):31-49, 2003.
  - C. Liu, D. Grenier, A.-L. Jousselme, É. Bossé, Reducing algorithm complexity for computing an aggregate uncertainty measure, IEEE Transactions on Systems, Man and Cybernetics-Part A: Systems and Humans 37: 669-679, 2007.
  - V.-N. Huynh, Y. Nakamori, Notes on "Reducing Algorithm Complexity for Computing an Aggregate Uncertainty Measure", IEEE Transactions on Cybernetics-Part A: Systems and Humans 40: 205-209, 2010.
  - M. Grabisch. Belief functions on lattices. International Journal of Intelligent Systems, 24:76-95, 2009.