ReBaStaBa : 
*handling Bayesian Network with R*

Jean-Baptiste.Denis@Jouy.Inra.Fr
ML.Delignette@Vet-Lyon.Fr
RPouillot@yahoo.Fr
PLAN

- Bayesian Network
- Defining a Bayesian Network [/bn/]
- Computing with a /bn/
- Some Features of rebastaba
(1) A set of random variates

- travel to Asia?
- Smoker?
- Tuberculosis?
- lung Cancer?
- Bronchitis
- T or C?
- bad X-ray?
- bad Dysphnae test?
(2) A set of random variates with conditional independence structure

\[ [A, S, T, C, B, TC, X, D] \]

(based on accepted causality)
(3) A set of random variates with conditional independence structure and associated probability distributions.

\[ P(A = y) = 0.01 \]
\[ P(C = y | S = n) = 0.01 \]
\[ P(C = y | S = y) = 0.10 \]
\[ P(TC = y | T = y, C = n) = 1 \]
\[ \ldots \ldots \]
Bayesian network

/Bn/s are a way to easily define a parsimonious joint probability distribution over a set of random variates.

(the graph is very attractive)
Defining a /bn/

<<Asian>>
role= Just for illustration for UseR'09
<<A>>
ltype= numcat
lpod= y n
lpara(p)= 0.01 0.99
<<S>>
ltype= numcat
lpod= y n
lpara(p)= 0.50 0.50
<<T>>
ltype= numcat
lpod= y n
lparent= A
lpara(p)= 0.01 0.05 0.99 0.95
<<B>>
ltype= numcat
lpod= y n
lparent= S
lpara(p)= 0.60 0.30 0.40 0.70

Contents of a text file named asia.dat
Defining a /bn/

Sourcing
asia.r

```r
bn <- file2bn("asia.dat");

# print(bn,"n");
# print(bn,"l","A");
# print(bn,"l","T");

dn <- bn2dn(bn, 10000);
print(dn@df[1:10,]);
print(table(dn@df[,3:4]));
print(grappa4mar2(bn, c("T","C")));
```

UseR'09
09_07_09
8/17
bn <- file2bn("asia.dat");

print(bn,"n");

##

### Node List

<table>
<thead>
<tr>
<th>nb.Var</th>
<th>ltype</th>
<th>Parent(s)</th>
<th>Node</th>
<th>Child(ren)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>numcat</td>
<td>{}</td>
<td>(A)-&gt;</td>
<td>{T}</td>
</tr>
<tr>
<td>2</td>
<td>numcat</td>
<td>{}</td>
<td>(S)=&gt;2</td>
<td>{C;B}</td>
</tr>
<tr>
<td>3</td>
<td>numcat</td>
<td>{A}</td>
<td>-&gt;(T)-&gt;</td>
<td>{TC}</td>
</tr>
<tr>
<td>4</td>
<td>numcat</td>
<td>{S}</td>
<td>-&gt;(C)-&gt;</td>
<td>{TC}</td>
</tr>
<tr>
<td>5</td>
<td>numcat</td>
<td>{S}</td>
<td>-&gt;(B)-&gt;</td>
<td>{D}</td>
</tr>
<tr>
<td>6</td>
<td>numcat</td>
<td>{T;C}</td>
<td>2=&gt;(TC)=&gt;2</td>
<td>{X;D}</td>
</tr>
<tr>
<td>7</td>
<td>numcat</td>
<td>{TC}</td>
<td>-&gt;(X)</td>
<td>{}</td>
</tr>
<tr>
<td>8</td>
<td>numcat</td>
<td>{B;TC}</td>
<td>2=&gt;(D)</td>
<td>{}</td>
</tr>
</tbody>
</table>
Computing with a bn/

Sourcing

asia.r

```r
#
print(bn,"n");
#
print(bn,"l","A");
print(bn,"l","T");
#
dn <- bn2dn(bn,10000);
```

probabilities are multiplied by 10^2

```
1 A: y n
2 : 1 99
```

probabilities are multiplied by 10^2

```
1 T: y n 
2 A --- --- ---
3 y : 1 99 
4 n : 5 95 
```

Computing with a `/bn/

Sourcing
asia.r

```r
print(bn,"l","T");
#
dn <- bn2dn(bn,10000);
print(dn@df);
print(table(dn@df[,3:4]));
print(grappa4mar2(bn,c("T","C")));
#
```

```
A  S  T  C  B  TC  X  D  >?<
1  n  y  n  n  n  y  n  n  n  0
2  n  y  n  n  y  n  n  n  0
3  n  n  n  n  y  n  n  n  0
4  n  n  n  n  n  n  n  n  0
5  n  y  n  n  n  n  n  n  0
6  n  y  n  n  y  n  n  n  0
7  n  y  n  n  y  n  n  n  0
8  n  n  n  n  n  n  n  n  0
9  n  n  n  n  n  n  n  n  0
10  ......................
```
Computing with a /bn/

Sourcing

```r
print(dn@df);
print(table(dn@df[,3:4]));
print(grappa4mar2(bn,c("T","C")));
#
```

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>y</td>
<td>n</td>
</tr>
<tr>
<td>T</td>
<td>y</td>
<td>n</td>
</tr>
<tr>
<td></td>
<td>26</td>
<td>487</td>
</tr>
</tbody>
</table>

``` empirical frequency table
Marginal dimensions are: T(2 categories) C(2 categories)
NO Conditional dimension
NO Conditioned dimension
probabilities are multiplied by 10^3
```

```
V1 V2 V3 V4 V5
1 C: y y n n
2 T: y n y n
3 : 3 52 47 898
```

``` empirical frequency table
```
Some Features

Main S4 Objects

/bn/ : Bayesian network

/gn/ : graph structure

/dn/ : specialized data frame

/alk/ : asked node
Some Features

/alk/ : main slots

<table>
<thead>
<tr>
<th>Feature</th>
<th>lpara</th>
<th>lrep</th>
<th>lnat</th>
<th>lvar</th>
<th>lparent</th>
<th>lpod</th>
</tr>
</thead>
<tbody>
<tr>
<td>normal</td>
<td>YES</td>
<td>yes</td>
<td>NO</td>
<td>NO</td>
<td>no</td>
<td>YES</td>
</tr>
<tr>
<td>uniform</td>
<td>YES</td>
<td>yes</td>
<td>NO</td>
<td>NO</td>
<td>no</td>
<td>YES</td>
</tr>
<tr>
<td>Bernoulli</td>
<td>YES</td>
<td>yes</td>
<td>NO</td>
<td>NO</td>
<td>no</td>
<td>NO</td>
</tr>
<tr>
<td>binomial</td>
<td>YES</td>
<td>yes</td>
<td>NO</td>
<td>NO</td>
<td>no</td>
<td>YES</td>
</tr>
<tr>
<td>Dirac</td>
<td>YES</td>
<td>yes</td>
<td>yes</td>
<td>NO</td>
<td>no</td>
<td>YES</td>
</tr>
<tr>
<td>multinomial</td>
<td>YES</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>YES</td>
</tr>
<tr>
<td>Dirichlet</td>
<td>YES</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>YES</td>
</tr>
<tr>
<td>numcat</td>
<td>YES</td>
<td>-</td>
<td>NO</td>
<td>yes</td>
<td>yes</td>
<td>YES</td>
</tr>
<tr>
<td>parcat</td>
<td>YES</td>
<td>-</td>
<td>NO</td>
<td>yes</td>
<td>no</td>
<td>YES</td>
</tr>
<tr>
<td>score</td>
<td>YES</td>
<td>-</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>easyp</td>
<td>YES</td>
<td>yes</td>
<td>YES</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>empidata</td>
<td>-</td>
<td>no</td>
<td>YES</td>
<td>YES</td>
<td>yes</td>
<td>YES</td>
</tr>
<tr>
<td>popula</td>
<td>-</td>
<td>no</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>program</td>
<td>-</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>yes</td>
<td>YES</td>
</tr>
</tbody>
</table>
Some Features

/alk/ : main slots

#-----------------------------
<<X>>
  ltype = normal
  lpod = 10 20
  lpara(mu) = 12
  lpara(sigma) = 1
#-----------------------------
<<Y>>
  ltype = normal
  lpod = 0 8
  lpara(mu) = sqrt({X}+1)
  lpara(sigma) = 2
#-----------------------------
# X is the parent of Y
# because the expectation of Y
# depends on it.
Some Features

Programming

11400 lines of codes
9283 comment lines (self generation of Rd documentation)

15 S4 classes are defined

229 functions are available

intensive checking with specialized functions
'check3rbsb' was called 129 times
'check4tyle' was called 73 times
'erreur' was called 402 times
'rapport' was called 23 times
Some Features

Near future

- Soon on http://riskassessment.r-forge.r-project.org/
- Enhancement on **Bugs (Jags)** transcription
- Graph analysis (detection of conditional independences)
- Interface with **deal**
- ......
Thanks for your attention