ReBaStaBa :

handling Bayesian Network with R

Jean-Baptiste.Denis@Jouy.Inra.Fr

ML.Delignette@Vet-Lyon.Fr

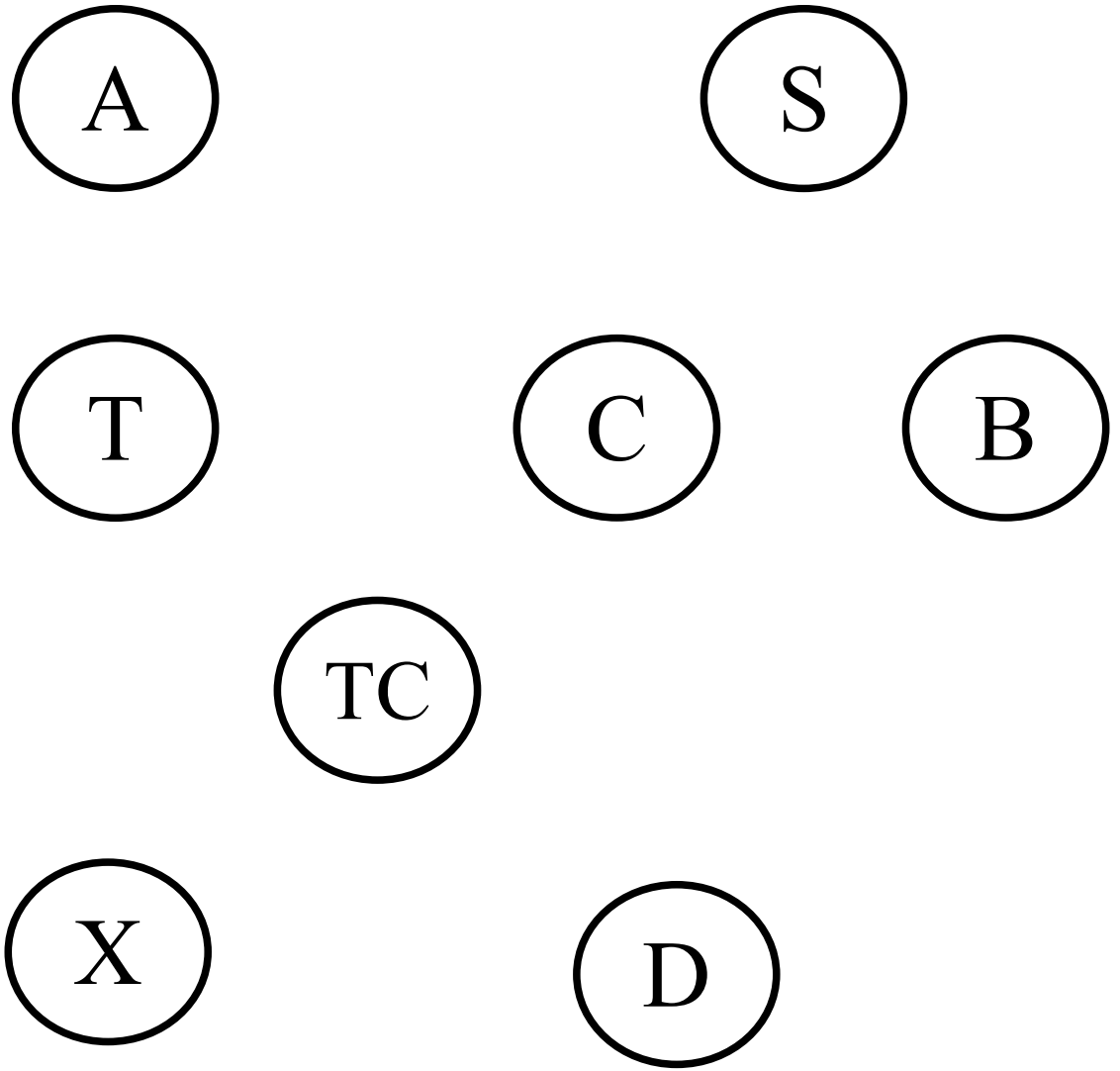
RPouillot@yahoo.Fr

PLAN

- Bayesian Network
- Defining a Bayesian Network [/bn/]
- Computing with a /bn/
- Some Features of rebastaba

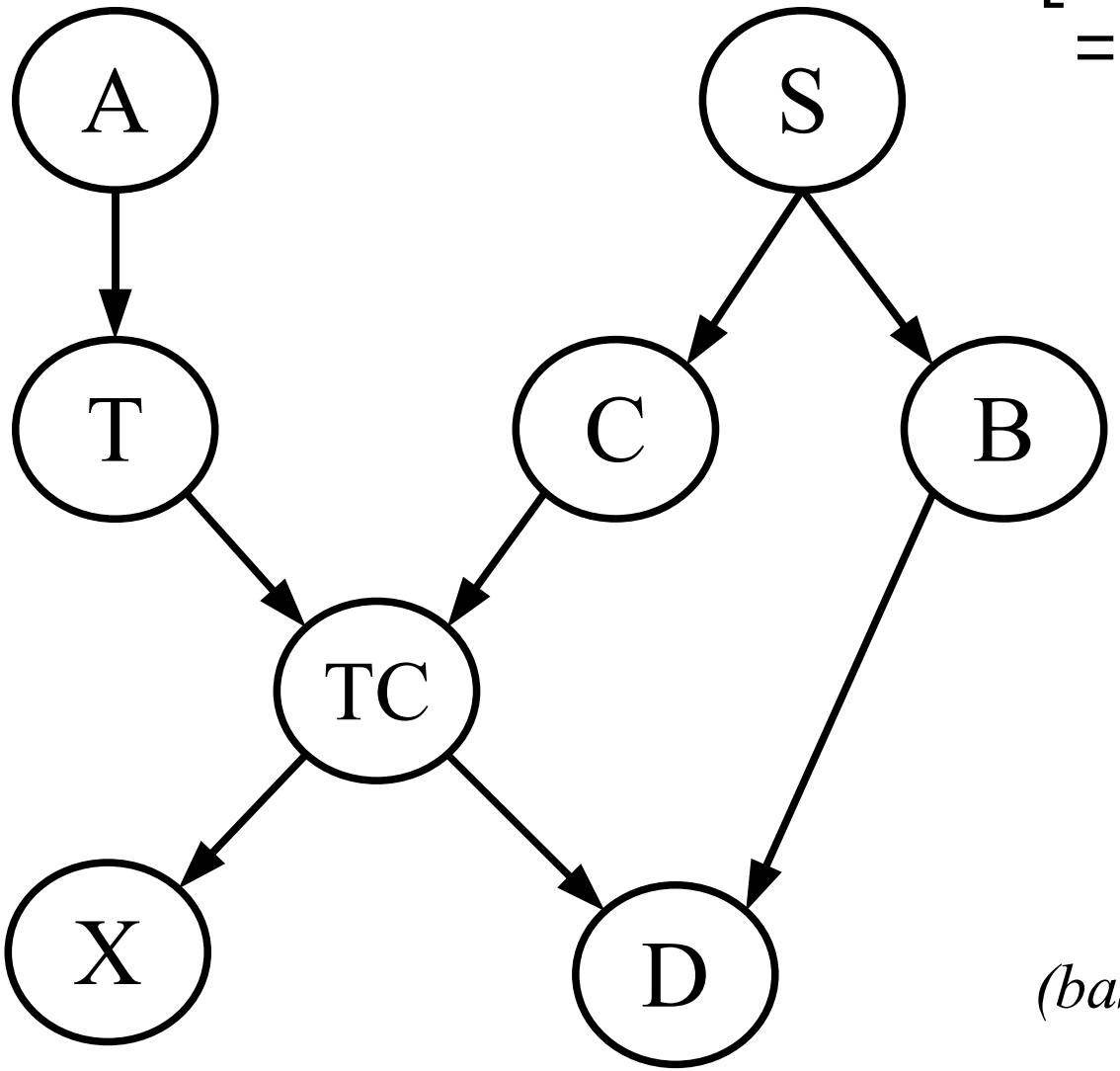
Bayesian network

(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

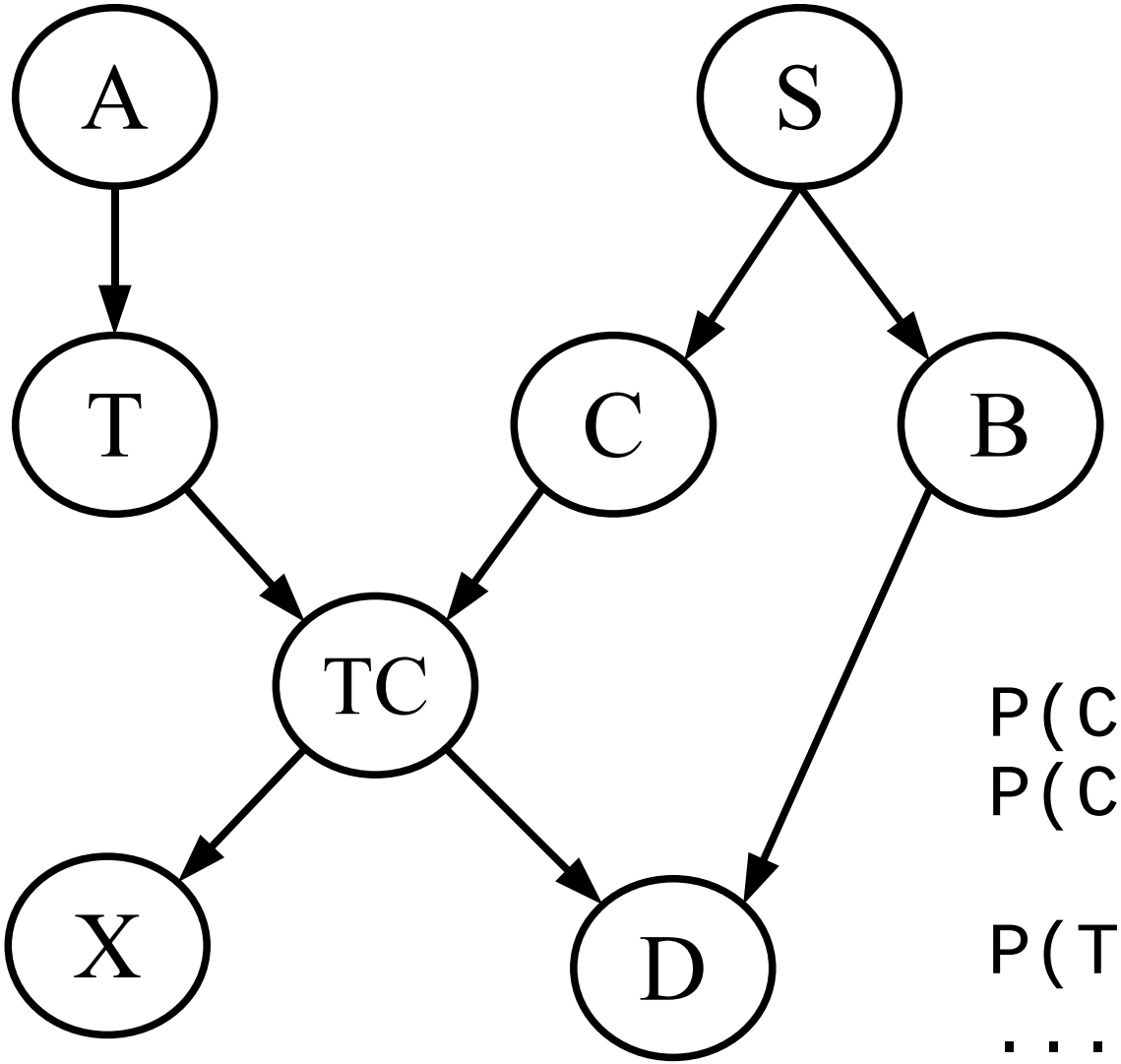



```
[A,S,T,C,B,TC,X,D]
 = [A][S][T|A]
   [C|S][B|S]
   [TC|T,C][X|TC]
   [D|TC]
```

(based on accepted causality)

Bayesian network

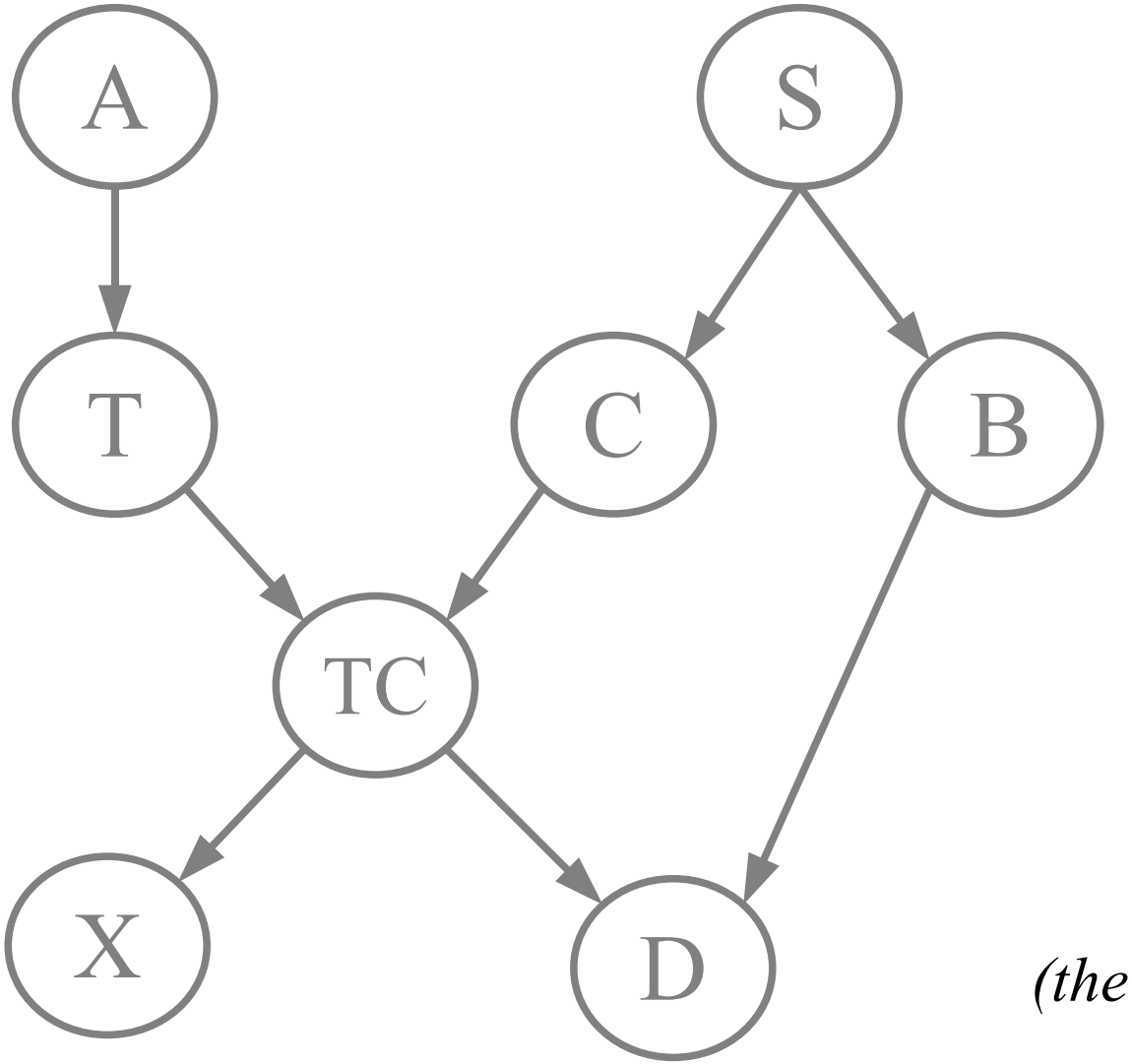
(3) A set of random variates with conditional independence structure and associated probability distributions.




```
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
.....
```

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/

Contents of a text file named **asia.dat**

```
<<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
```

Defining a /bn/

Sourcing
asia.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")));
#
```

Computing with a /bn/

Sourcing
asia.r

```
#
bn <- file2bn("asia.dat");
#
print(bn,"n");
#
```

```
   Node List
   =========
  nb.Var  ltype Parent(s)      Node   Child(ren)
1      1 numcat        {}      (A)->          {T}
2      1 numcat        {}      (S)=>2       {C;B}
3      1 numcat       {A}   ->(T)->         {TC}
4      1 numcat       {S}   ->(C)->         {TC}
5      1 numcat       {S}   ->(B)->          {D}
6      1 numcat     {T;C} 2=>(TC)=>2       {X;D}
7      1 numcat      {TC}   ->(X)            {}
8      1 numcat    {B;TC}  2=>(D)            {}
```

Computing with a /bn/

Sourcing
asia.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

```
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 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/

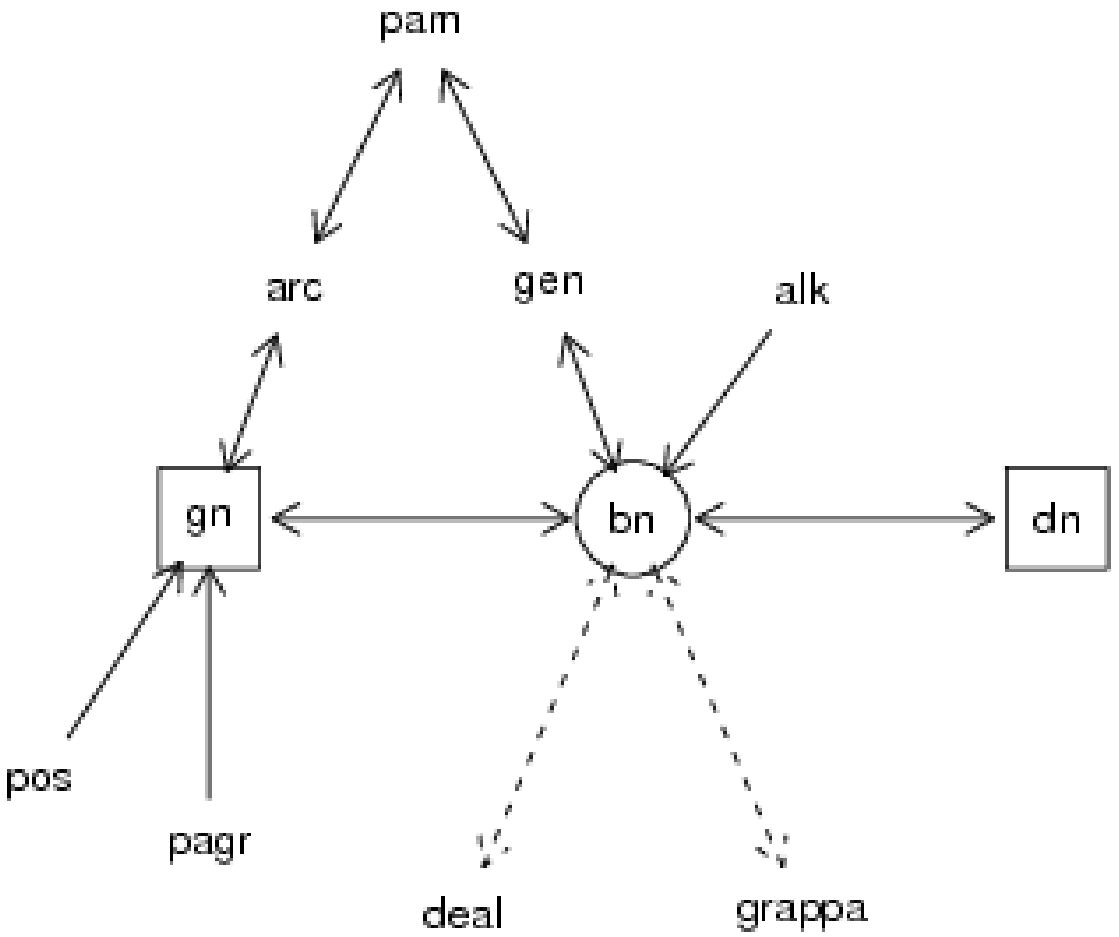
Sourcing **asia.r**

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

```
   C
T      y    n
  y   26  487
  n  499 8988
>>> --------------->  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
>>> --------------->  Computed frequency table
```

Some Features

Main S4 Objects



/bn/ : Bayesian network

/gn/ : graph structure

/dn/ : specialized data frame

/alk/ : asked node

/alk/ : main slots

```
<YES> for 'must be provided by the user'
<yes> for 'can be provided by the user but there is a default value'
< no> for 'necessary but generated by rebastaba'
< NO> for 'necessary but fixed by rebastaba'
< - > for 'irrelevant'
```

	lpara	lrep	lnat	lvar	lparent	lpod
normal	**YES**	**yes**	**NO**	**NO**	**no**	**YES**
uniform	YES	yes	NO	NO	no	YES
Bernoulli	YES	yes	NO	NO	no	NO
binomial	YES	yes	NO	NO	no	YES
Dirac	YES	yes	yes	NO	no	YES
multinomial	YES	no	no	yes	no	YES
Dirichlet	YES	no	no	yes	no	YES
numcat	**YES**	**-**	**NO**	**yes**	**yes**	**YES**
parcat	YES	-	NO	yes	no	YES
score	YES	-	NO	NO	YES	YES
easyp	YES	yes	YES	yes	no	YES
empidata	-	no	YES	YES	yes	YES
popula	-	no	YES	YES	NO	YES
program	**-**	**YES**	**YES**	**YES**	**yes**	**YES**

Some Features

/alk/ : main slots

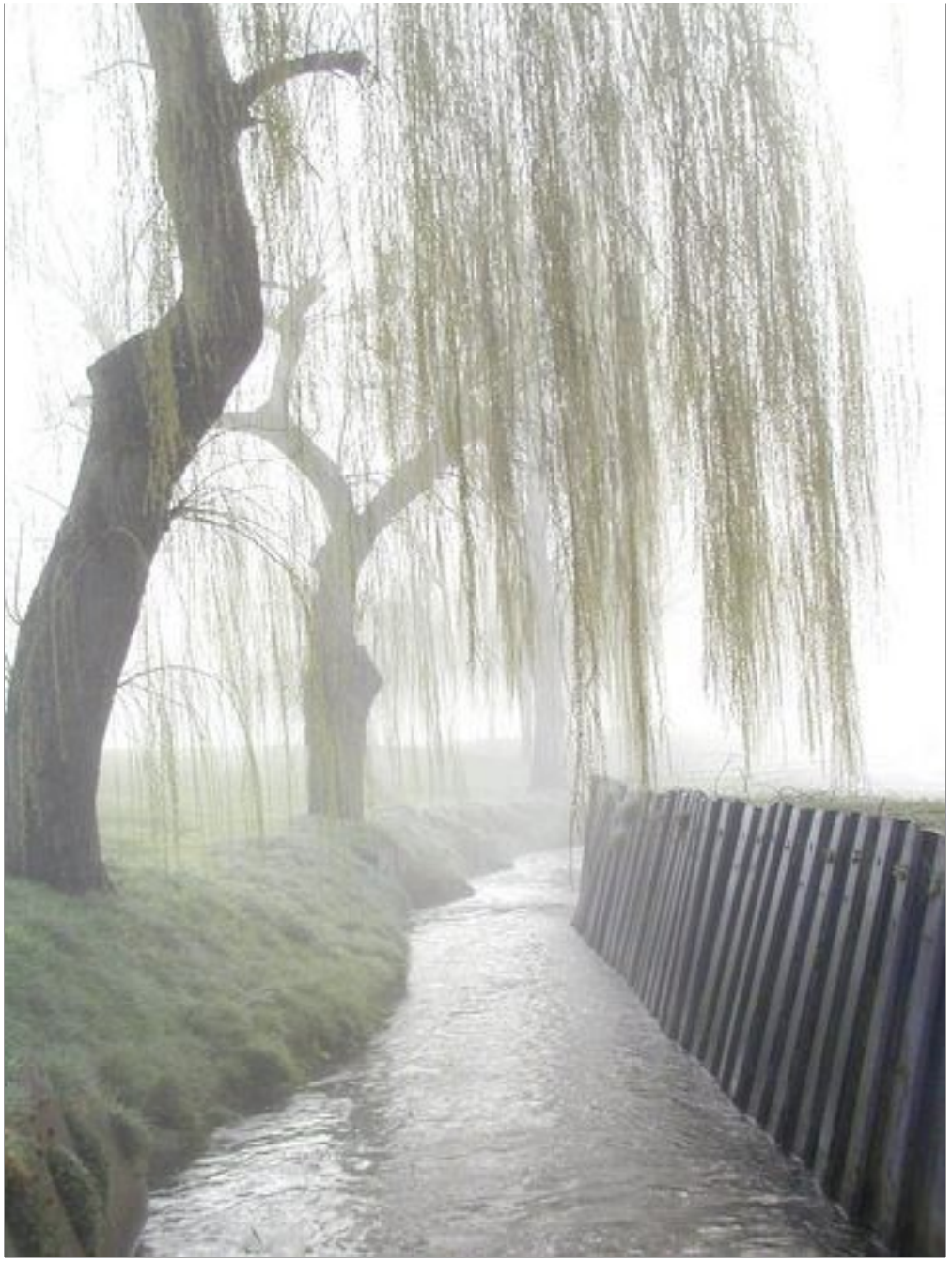
```
#-------------------------------
<<X>>
ltype= normal
lpod= 10 20
lpara(mu)= 12
lpara(sigma)= 1
#-------------------------------
<<Y>>
ltype= normal
lpod= 0 8
lpara(mu)= sqrt({{X}}+1)
lpara(sigma)= 2
#-------------------------------
# X is the parent of Y
# because the expectation of Y
# depends on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

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
```

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