Julie Jacques

METHODES D'ESTIMATION DE LA FONCTION DE DEPENDANCE DES COPULES DE VALEURS EXTREMES.

Mémoire
présenté
à la Faculté des études supérieures
de l'Université Laval
pour l'obtention
du grade de maître ès sciences (M.Sc.)

Département de mathématiques et de statistique
FACULTÉ DES SCIENCES ET DE GÉNIE
UNIVERSITÉ LAVAL

FEVRIER 1998

0-612-26221-9

Résumé

Pour comprendre le comportement des valeurs extrêmes, il est nécessaire de s'intéresser à leur modélisation. Lorsque les valeurs extrêmes sont unidimensionnelles, l'ensemble de leurs lois possède une structure paramétrique. Par contre, lorsque les valeurs extrêmes sont bidimensionnelles, l'ensemble des lois ne possède pas de structure paramétrique. Ces lois sont caractérisées par leurs marginales et par une fonction appelée fonction de dépendance de la loi bivariée des valeurs extrêmes. Plusieurs méthodes paramétriques et non paramétriques ont été proposées pour estimer cette fonction. Dans ce mémoire, nous nous intéressons principalement à trois de ces méthodes non paramétriques. Nous comparerons ces estimateurs grâce à différentes simulations qui seront effectuées ainsi qu'à l'analyse d'un ensemble de données.

Remerciements

C'est avec un immense plaisir que je désire exprimer ma gratitude à mon directeur de recherche, le professeur Philippe Capéraà qui, grâce à sa patience, son encouragement, sa disponibilité et ses nombreuses connaissances, m'a permis de réaliser ce mémoire. Je tiens aussi à le remercier du financement qu'il m'a fourni par l'intermédiaire de subventions octroyées par le fonds FCAR du Québec.

Je voudrais souligner la précieuse collaboration de madame Anne-Laure Fougères, maître de conférence à l'Université Paul-Sabatier de Toulouse, en particulier pour les nombreux conseils ainsi que pour les programmes informatiques qu'elle a bien voulu mettre à ma disposition.

Je désire également remercier madame Louise Rémillard, de l'Alcan, ainsi que monsieur Luc Perreault, de l'INRS, qui m'ont fourni et autorisé à utiliser des données de l'Alcan.

Enfin qu'il me soit permis d'avoir une pensée pour les membres de ma famille, mes amis, et plus particulièrement mon conjoint François, qui ont su m'encourager tout au long de ce projet.

Table des matières

Liste des annexes

Liste des figures

Liste des tableaux

Chapitre 1

Introduction

Dans plusieurs domaines, tels l'hydrologie, l'étude de problèmes environnementaux, la fiabilité, etc., les événements auxquels on s'intéresse font intervenir des observations extrêmes, que ce soit des maxima ou des minima. Par exemple, lors de la construction d'un barrage hydroélectrique sur une rivière, il est nécessaire de s'intéresser à la hauteur maximale que peut atteindre le cours d'eau. De la même façon, lorsqu'on étudie la fiabilité de certains appareils, les observations sur les pièces les plus fragiles sont des données très importantes. Cependant, dans de nombreux cas, les données sont multidimensionnelles et les évènements extrêmes d'intérêt montrent certaines formes de dépendance qu'il est utile de pouvoir modéliser. Ces données peuvent être, soit directement des maxima ou des minima, soit des données multidimensionnelles dont on cherche à modéliser la loi des valeurs extrêmes correspondantes. Nous limiterons ici notre étude au cas bivarié.

Lorsque les observations sont unidimensionnelles, la loi des valeurs extrêmes est complètement définie par trois paramètres réels. Par contre, dans le cas bivarié, la loi conjointe des extrêmes est caractérisée par les lois marginales et par une fonction convexe, définie sur $[0, 1]$, appelée fonction de dépendance de la loi bivariée des valeurs extrêmes. Plusieurs auteurs se sont intéressés à l'estimation de cette fonction de dépendance.

Certains ont utilisé une démarche basée sur différentes approches paramétriques, par exemple Tawn (1988), Tiago de Oliveira (1984), etc. D'autres auteurs ont proposé des estimateurs de la fonction de dépendance qui s'appuient sur une démarche non paramétrique. Citons par exemple Pickands (1981), Joe, Smith et Weissman (1992), Khoudraji (1995), ainsi que Capéraà, Fougères et Genest (1997a).

Le principal objectif de ce travail est de comparer certaines méthodes d'estimation non paramétriques de la fonction de dépendance, ainsi que leurs comportements pour de petits échantillons.

Après avoir présenté la loi des valeurs extrêmes ainsi que quelques résultats pour le cas univarié et bivarié au chapitre 2, nous exposerons au chapitre 3 quelques estimateurs non paramétriques de la fonction de dépendance. Le premier, qui est conçu pour des observations extrêmes, est une extension de celui de Capéraà et al. (1997a) au cas où les marges de la loi de ces observations ne sont pas connues. Les deux autres estimateurs considérés dans notre étude sont ceux de Khoudraji (1995) et de Joe et al. (1992). Au chapitre 4, nous aborderons deux analyses différentes pour comparer ces méthodes d'estimation. Dans un premier temps, nous comparerons les propriétés de ces estimateurs calculés à partir d'échantillons de lois bivariées. Dans un deuxième temps, ces comparaisons seront effectuées sur les mêmes estimateurs mais ceux-ci seront calculés à partir d'échantillons de valeurs extrêmes. Dans le dernier chapitre, nous appliquerons ces estimateurs à des données concernant des réservoirs d'eau utilisés pour la production d'électricité nécessaire à l'entreprise Alcan, et nous en comparerons les résultats.

Chapitre 2

Lois de valeurs extrêmes

Dans plusieurs domaines d'application, il est nécessaire de s'intéresser à la notion de valeurs extrêmes. La théorie des valeurs extrêmes a pour objet la modélisation et l'étude de problèmes où les observations sont des valeurs maximums ou minimums d'un certain échantillon. Il existe dans le cas univarié une vaste littérature sur le sujet. On peut citer par exemple les tous premiers travaux effectués par L. von Bortkiewicz (1922) et R. von Mises (1923). Ce sujet fut également étudié par Fisher et Tippett (1928) qui établirent les trois lois limites des valeurs extrêmes, et par R.von Mises (1954) qui proposa la notion de domaine d'attraction.

Contrairement au cas univarié, les travaux effectués dans le cas bivarié sont nettement moins nombreux. Parmi les auteurs qui se sont intéressés au cas bivarié, on peut citer Tiago de Oliveira (1984), Pickands (1981), Deheuvels (1984), Resnick (1987) et Galambos (1987).

Cette section est consacrée à la présentation de la loi des valeurs extrêmes ainsi que de quelques résultats classiques. Dans un premier temps, nous présenterons le cas univarié pour ensuite étendre ces définitions au cas bivarié.

2.1 Cas univarié

Considérons un échantillon $X_1, ...X_n$ d'une loi F définie sur R, et posons

$$Z_n = max(X_1, ..., X_n),$$

c'est-à-dire Z_n est la plus grande observation de l'échantillon. S'il existe deux suites de réels $a_n > 0$ et b_n telles que

$$\lim_{n\to\infty} P(\frac{Z_n - b_n}{a_n} \leq x) = \lim_{n\to\infty} F^n(a_n x + b_n) = M(x), \tag{2.1}$$

où M est une fonction de répartition qui n'est pas dégénérée, on dit que M est une loi des valeurs extrêmes du maximum.

Si une fonction de répartition G est telle qu'il existe deux constantes $\alpha > 0$ et β telles que $G(x) = M(\alpha x + \beta)$, on dit que G est du même type que M.

Fisher et Tippett ont montré en 1928 que la fonction de répartition d'une loi des valeurs extrêmes associées au maximum appartenait à l'un des trois types suivants:

$$\begin{gathered} \Lambda(x) = exp\{-exp(-x)\}, \ -\infty < x < \infty, \\ \Phi_a(x) = exp(-x^{-a}), \ x > 0, \ a > 0, \\ \Psi_a(x) = exp\{-(-x)^a\}, \ x \leq 0, \ a > 0. \end{gathered} \tag{2.2}$$

Ces lois sont appelées respectivement, loi de Gumbel, loi de Fréchet et loi de Weibull standards. $\Phi_1(x)$ et $\Psi_1(x)$ sont les lois unités de Fréchet et de Weibull respectivement. Notons que si M n'est pas sous une de ces trois formes, aucune constante a_n et b_n ne satisfont (2.1).

Ces trois types de lois peuvent se mettre sous la forme d'une famille paramétrique appelée famille des distributions des valeurs extrêmes généralisées de von Mises (1954) définie par

$$M(x; \mu, \sigma, k) = exp[-\{1 - k(x - \mu)/\sigma\}^{1/k}],$$

où $\sigma > 0$, $\mu, k \in R$ et pour tout x tel que $1 - k(x - \mu)/\sigma > 0$. Ainsi, lorsque k tend vers zéro, nous retrouvons une loi de Gumbel, pour k positif, nous avons une loi de Weibull et enfin, pour k négatif, une loi de Fréchet.

On peut définir de façon similaire la loi des valeurs extrêmes du minimum; si

$$W_n = min(X_1, ..., X_n).$$

est la plus petite observation de l'échantillon et s'il existe deux suites de réels $a_n > 0$ et b_n telles que

$$\lim_{n\to\infty} P(\frac{W_n - b_n}{a_n} \leq x) = \lim_{n\to\infty} \{1 - F(a_n x + b_n)\}^n = L(x),$$

où L est une fonction de répartition qui n'est pas dégénérée, on dit que L est une loi des valeurs extrêmes du minimum. Dans ce cas, L est de l'un des trois types suivants:

$$\begin{aligned} \Lambda^*(x) &= 1 - exp\{-exp(x)\}, \ -\infty < x < \infty, \\ \Phi_a^*(x) &= 1 - exp\{-(-x)^{-a}\}, \ x < 0, \ a > 0, \\ \Psi_a^*(x) &= 1 - exp(-x^a), \ x \geq 0, \ a > 0. \end{aligned} \tag{2.3}$$

Ce résultat s'obtient aisément en remarquant que la loi du minimum d'un échantillon d'une loi F se déduit de la loi du maximum d'un échantillon de la loi symétrique de F par rapport à l'origine. Notons encore qu'une loi quelconque de (2.2) et (2.3) peut s'obtenir à partir d'une des autres par une simple transformation de la variable aléatoire X. Par exemple, si X suit la loi de Gumbel standard Λ, alors $Y = exp(-X)$ suit la loi exponentielle Ψ_1^*.

L'existence des constantes a_n et b_n dans la définition d'une loi des valeurs extrêmes repose en fait sur des propriétés vérifiées par la loi F. Des conditions nécessaires et suffisantes sur F ont été obtenues pour l'existence de la loi limite M (voir par exemple Galambos, 1987, pp. 53-54 et 75-76). Lorsque cette limite existe, on dit alors que la loi F est dans le domaine d'attraction de M ou que M est l'attracteur de F.

Exemple 2.1: *Soit $X_1, ..., X_n$ un échantillon d'une loi F continue. Considérons les variables aléatoires $Y_i = 1/\{1 - F(X_i)\}$, $i = 1, ..., n$. Soit G la fonction de répartition définie par*

$$G(x) = P(Y_i \leq x) = \begin{cases} 0 & si\ x < 1 \\ \\ 1 - \frac{1}{x} & si\ x \geq 1\,. \end{cases}$$

On vérifie facilement que sa fonction de survie $1-G$ *est à variations régulières d'exposant* -1*, c'est-à-dire, pour* $x \geq 1$,

$$\lim_{t\to\infty} \frac{1-G(tx)}{1-G(t)} = x^{-1}$$

(voir par exemple Resnick, 1987, p.12). On déduit alors du Théorème 2.1.1 de Galambos (1987) que la loi limite du maximum des Y_i *existe et qu'elle est une loi de Fréchet, les constantes* a_n *et* b_n *étant respectivement égales à* n *et 0.*

2.2 Cas bivarié

Comme pour le cas univarié, il est nécessaire dans le cas bivarié d'ordonner les couples de variables aléatoires. Cependant, puisqu'il est plutôt rare que les deux composantes d'un même couple soient des maxima ou des minima sur l'ensemble des composantes, on doit alors déterminer en quel sens les observations seront ordonnées. Contrairement au cas unidimensionnel, pour lequel le maximum ne pouvait être déterminé qu'en utilisant les définitions sur les statistiques d'ordre, il existe plusieurs choix possibles pour définir le concept d'ordre dans le cas bivarié. Selon Resnick (1987), chapitre 5, le choix communément adopté consiste à ordonner les vecteurs composante par composante, c'est-à-dire que le maximum sur la première composante des couples ainsi que le maximum sur la deuxième composante sont retenus. Notons que le couple formé de ces deux maxima n'est pas en général une observation de l'échantillon. En effet, si l'on examine la Figure 2.1 représentant les couples d'un échantillon bivarié, on s'aperçoit que le point qui représente le couple de maxima n'est pas une observation de l'échantillon. C'est cette principale différence qui rend difficile la généralisation du cas univarié au cas bivarié.

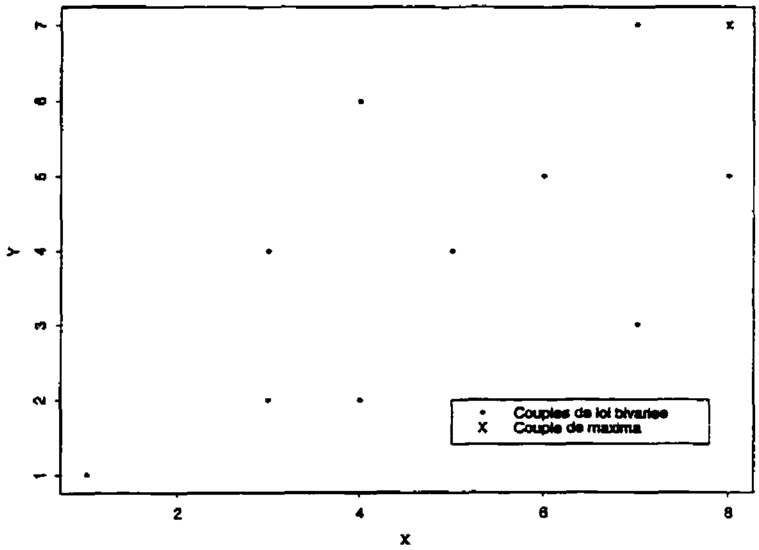


Figure 2.1: *Couples de loi bivariée ainsi que le couple de maxima sur les 2 composantes.*

Considérons l'échantillon bivarié $(X_1, Y_1), ..., (X_n, Y_n)$ de fonction de répartition F définie sur R^2, et posons

$$Z_{1n} = max(X_1, ..., X_n)$$

et

$$Z_{2n} = max(Y_1, ..., Y_n),$$

c'est-à-dire que Z_{1n} est la plus grande observation de l'échantillon $X_1, ..., X_n$; et Z_{2n} celle de l'échantillon $Y_1, ..., Y_n$. S'il existe des suites de réels $a_{1n}, a_{2n} > 0$ et b_{1n}, b_{2n} telles que

$$\lim_{n\to\infty} P(\frac{Z_{1n} - b_{1n}}{a_{1n}} \leq x, \frac{Z_{2n} - b_{2n}}{a_{2n}} \leq y) = M(x, y), \tag{2.4}$$

où M est une fonction de répartition bivariée dont les lois marginales ne sont pas dégénérées, alors M est appelée loi bivariée des valeurs extrêmes du maximum.

Comme pour le cas univarié, on dit alors que la fonction de répartition F appartient au domaine d'attraction maximum de M.

De façon similaire, si

$$W_{1n} = min(X_1, ..., X_n)$$

et

$$W_{2n} = min(Y_1, ..., Y_n),$$

c'est-à-dire que W_{1n} est la plus petite observation de l'échantillon $X_1, ..., X_n$ et W_{2n} celle de l'échantillon $Y_1, ..., Y_n$, et qu'ils existent des suites de réels $a_{1n}, a_{2n} > 0$ et b_{1n}, b_{2n} telles que

$$\lim_{n\to\infty} P(\frac{W_{1n} - b_{1n}}{a_{1n}} \leq x, \frac{W_{2n} - b_{2n}}{a_{2n}} \leq y) = L(x, y),$$

où L est une fonction de répartition bivariée, dont les lois marginales ne sont pas dégénérées, alors L est appelée loi bivariée des valeurs extrêmes du minimum.

Nous nous limiterons par la suite à exposer les résultats pour les lois des valeurs extrêmes du maximum seulement. Le théorème suivant (voir par exemple Galambos, 1987, p.293) précise quelques propriétés des lois bivariées des valeurs extrêmes.

Théorème 2.1. *Toute loi limite M vérifiant (2.4) est continue. De plus, ses lois marginales admettent un des trois types énumérés en (2.2).*

Puisque ces fonctions de répartition M sont continues, elles peuvent s'écrire de façon unique sous la forme

$$M(x, y) = C\{F(x), G(y)\},$$

où $C(u, v)$ est une fonction de répartition bidimensionnelle appelée copule (Sklar, 1959), définie sur le carré unité, et dont les lois marginales sont uniformes. Cette forme permet de séparer l'effet des marges de l'effet de la dépendance du couple (X, Y).

Théorème 2.2. *Soit $(X_1, Y_1), ..., (X_n, Y_n)$ un échantillon bidimensionnel d'une loi F. Alors, il existe des constantes réelles $a_{1n}, a_{2n} > 0$ et b_{1n}, b_{2n} telles que*

$$\lim_{n\to\infty} P(\frac{Z_{1n} - b_{1n}}{a_{1n}} \leq x, \frac{Z_{2n} - b_{2n}}{a_{2n}} \leq y) = M(x, y),$$

où M est une fonction de répartition qui n'est pas dégénérée, si et seulement si les lois marginales de M sont des lois de valeurs extrêmes univariées et si la copule C qui lui est associée satisfait pour tout $k \in N\backslash\{0\}$, et pour tous $u, v \in [0, 1]$, l'équation

$$C(u, v) = C^k(u^{1/k}, v^{1/k}).$$

Ce dernier résultat, qui est une caractérisation de loi des valeurs extrêmes bivariée, ne fournit pas la forme générale de ces lois. Il existe d'autres caractérisations mettant en lumière cette forme qui, naturellement, dépend des lois marginales. Cependant, comme on l'a vu dans la section précédente, il est possible d'obtenir une loi des valeurs extrêmes univariée à partir d'une autre par un simple changement de variables. Ainsi, peut-on choisir un des trois types de lois marginales sans affecter sensiblement la représentation d'une fonction de valeurs extrêmes bivariée M.

La représentation suivante, due à Pickands (1981), est donnée dans le cas d'une loi M bidimensionnelle admettant des marges de Fréchet unités (pour d'autres cas, voir par exemple Pickands, 1981, Galambos, 1987, p. 309, et Resnick, 1987, p. 268 et 272).

Théorème 2.3. *Soit M une loi bidimensionnelle dont les lois marginales sont des lois de Fréchet unités. Soit $S(p)$ une mesure positive finie sur l'intervalle* $[0,1]$. *Alors M est une loi des valeurs extrêmes bivariée si et seulement si, pour tout $x, y \in R^+$*

$$M(x,y) = exp[-\int_0^1 max\{p/x, (1-p)/y\} dS(p)]$$

et

$$M(x,y) = 0, \text{ ailleurs.}$$

Remarquons que la contrainte imposée sur les lois marginales de M implique

$$\int_0^1 p dS(p) = \int_0^1 (1-p) dS(p) = 1,$$

d'où l'on déduit que $S/2$ est une loi de probabilité sur $(0,1)$ de moyenne $1/2$. Pickands (1981) a montré que la loi M pouvait alors s'écrire sous la forme suivante:

$$log M(x,y) = -(\frac{1}{x} + \frac{1}{y}) A(\frac{y}{x+y}), \tag{2.5}$$

où la fonction A, appelée fonction de dépendance, est liée à la loi $S/2$ par la relation suivante:

$$A(t) = \int_0^1 max\{pt, (1-p)(1-t)\} dS(p). \tag{2.6}$$

La fonction A est une fonction définie sur l'intervalle $[0,1]$, vérifiant $max(t, 1-t) \leq A(t) \leq 1$. Puisque A est une fonction convexe, $A'(t)$ existe partout, sauf peut-être en un nombre dénombrable de points, auquel cas les dérivées à gauche et à droite existent: $-1 \leq A'(t) \leq 1$, $\lim_{t\to 0^+} A'(t) \geq -1$, $\lim_{t\to 1^-} A'(t) \leq 1$. Enfin, $A(t)$ est symétrique par rapport à $1/2$ si et seulement si les variables aléatoires U et V sont échangeables, c'est-à-dire $C(u,v) = C(v,u)$. Notons que lorsque pour $0 \leq t \leq 1$, $A(t) = 1$ alors

$$M(x,y) = e^{-x^{-1}-y^{-1}},$$

ce qui correspond à l'indépendance entre X et Y.

De la relation (2.5), on déduit aisément que la copule $C(u,v)$ associée à la loi M a la forme suivante:

$$C(u,v) = exp[\{log(uv)\}A\{\frac{log(u)}{log(uv)}\}], \ \forall\ 0 \leq u,v \leq 1. \tag{2.7}$$

Cette copule ne dépend pas des lois marginales de la loi M et donc toutes les copules associées aux lois des valeurs extrêmes ont la forme (2.7). On les appelle les copules des valeurs extrêmes. Les exemples qui suivent présentent certaines copules de valeurs extrêmes ainsi que les graphiques représentant la fonction de dépendance associée à ces copules.

Exemple 2.2: La copule de Gumbel de type B

$$A(t) = \{(1-t)^r + t^r\}^{1/r}, \; pour \; r \geq 1;$$

$$C(u,v) = exp[\{-log(u)\}^r + \{-log(v)\}^r]^{1/r}.$$

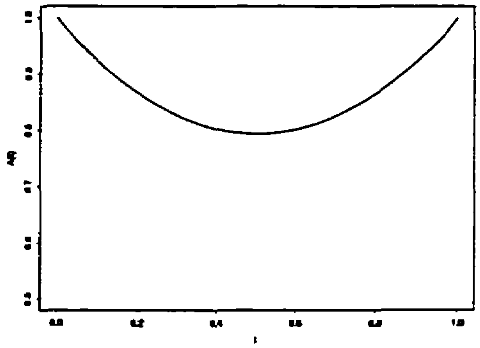


Figure 2.2: *Graphique de la fonction de dépendance A de la copule de Gumbel de type B pour $r = 1.5$.*

Exemple 2.3: La copule de Galambos

$$A(t) = 1 - \{(1-t)^{-\theta} + t^{-\theta}\}^{-1/\theta}, \; pour \; \theta \geq 0;$$

$$C(u,v) = uvexp[\{-log(u)\}^{-\theta} + \{-log(v)\}^{-\theta}]^{-1/\theta}.$$

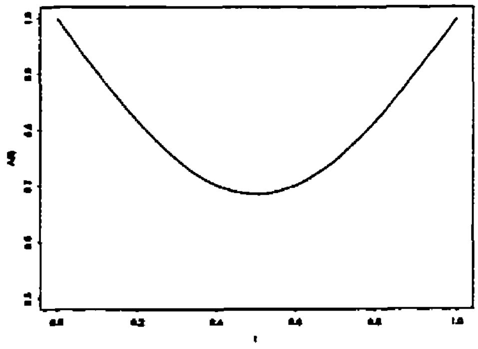


Figure 2.3: *Graphique de la fonction de dépendance A de la copule de Galambos pour $\theta = 1.5$.*

Exemple 2.4: La copule de Cuadras-Augé généralisée

$$A(t) = max\{1-\theta t, 1-\beta(1-t)\}, \ pour\ 0 \leq \theta, \beta \leq 1;$$

$$C(u,v) = u^{1-\theta}v^{1-\beta}min(u^{\theta}, v^{\beta}).$$

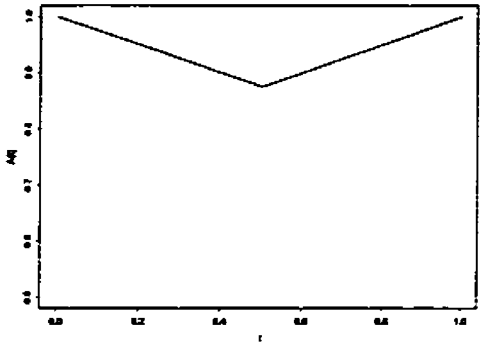


Figure 2.4: *Graphique de la fonction de dépendance A de la copule de Cuadras-Augé généralisée pour $\theta = 0.25$ et $\beta = 0.25$.*

Ainsi, lorsqu'une loi bivariée des valeurs extrêmes possède des marges connues, son expression ne dépend alors que de la fonction A. Dans le chapitre suivant, nous allons nous intéresser à l'estimation de cette fonction de dépendance A dans un cadre non paramétrique.

Chapitre 3

Estimation de la fonction de dépendance

Dans le chapitre précédent, nous avons décrit brièvement certaines caractérisations et représentations des lois bivariées des valeurs extrêmes liées aux maxima. Dans ce chapitre, nous allons nous intéresser à l'estimation de la fonction de dépendance de ces lois dans les deux situations suivantes.

Dans la première, on dispose d'un échantillon d'une loi bivariée de valeurs extrêmes. On rencontre cette situation en pratique lorsque, par exemple, on s'intéresse à des maxima annuels et que l'on possède des données bidimensionnelles journalières ou hebdomadaires, et ce sur plusieurs années. On peut alors, pour chaque ensemble de données d'une année, déterminer les valeurs extrêmes. Ainsi, possède-t-on un échantillon, dont la taille est égale au nombre d'années, que l'on peut considérer issu d'une loi de valeurs extrêmes bivariée.

Dans la deuxième situation, on dispose d'un échantillon d'une loi F bidimensionnelle et on cherche à estimer la loi asymptotique des maxima de ces données. Ceci se produit, par exemple, lorsqu'on ne dispose que d'un ensemble de données sur une année et que l'on s'intéresse aux maxima annuels.

Le premier estimateur que nous allons décrire, proposé par Capéraà, Fougères et Genest (1997a), est adapté à la première situation alors que les deux suivants, dûs à Khoudraji (1995) et à Joe, Smith et Weissman (1992), concernent la deuxième. Ces derniers s'appuient sur une approche de l'étude des lois des valeurs extrêmes multivariées basée sur les processus ponctuels (voir par exemple de Haan, 1985, ou Resnick, 1987, chapitre 5). Plus précisément, considérons un échantillon $(X_1, Y_1), ..., (X_n, Y_n)$ d'une loi bidimensionnelle F appartenant au domaine d'attraction d'une loi de valeurs extrêmes bivariée M. Supposons que les marges de F soient des lois de Fréchet unités, ce qui peut toujours s'obtenir par un simple changement de variables, et considérons le processus ponctuel sur R^{2+} associé à $\{\frac{1}{n}(X_1, Y_1), ..., \frac{1}{n}(X_n, Y_n)\}$, où la constante $1/n$ est due au fait que les marges sont de Fréchet (voir l'Exemple 2.1). Ce processus ponctuel converge en loi vers un processus de Poisson non-homogène sur $R^{+2}\backslash\{0,0\}$, dont la mesure d'intensité μ^*, ou mesure moyenne, vérifie, pour tout Borélien B de $R^{+2}\backslash\{0,0\}$ et pour tout t réel strictement positif,

$$\mu^*(tB) = t^{-1}\mu^*(B).$$

Posons $R_i = \frac{X_i+Y_i}{n}$ et $W_i = \frac{X_i}{nR_i}$, $i = 1, ...n$. Alors, avec ces nouvelles variables, la mesure d'intensité μ^* du processus limite s'exprime comme une mesure produit, soit

$$\mu^*(dr \text{ x } dw) = \frac{dr}{r^2}dS(w), \tag{3.1}$$

où S est une mesure positive finie sur $[0, 1]$ vérifiant les conditions

$$\int_0^1 wdS(w) = \int_0^1 (1-w)dS(w).$$

On déduit alors de la Proposition 5.11, chapitre 5 de Resnick (1987), que la relation entre μ^* et S est donnée par

$$-log\{M(x,y)\} = \int_0^1 max(\frac{w}{x}, \frac{1-w}{y})dS(w) = \mu^*(\{[0,x] \text{ x } [0,y]\}^c) = \mu(x,y),$$

d'où l'on tire d'après (2.5)

$$\mu(x,y) = (\frac{1}{x} + \frac{1}{y})A(\frac{y}{x+y}), \tag{3.2}$$

A étant la fonction de dépendance de la loi M.

Nous terminons cette section par un résultat de de Haan (1985) qui sera utile dans la méthode d'estimation de Khoudraji.

Proposition 3.1. *Soit (X,Y) un couple de variables aléatoires de loi F, appartenant au domaine d'attraction d'une loi H, dont les marginales sont des lois de Fréchet unités, alors*

$$\lim_{u\to\infty} P(\frac{X}{X+Y} \leq w \mid X+Y > u) = \frac{S(w)}{2}.$$

De cette proposition, on déduit que $S/2$ est la fonction de répartition conditionnelle de la variable $X/(X+Y)$, sachant que $X+Y$ est supérieure à u, et ce lorsque u tend vers l'infini.

Dans les sections suivantes, nous allons décrire les trois méthodes d'estimation de la fonction de dépendance d'une loi de valeurs extrêmes bidimensionnelle. Ces méthodes seront comparées dans un chapitre ultérieur.

3.1 Méthode d'estimation de Capéraà, Fougères et Genest

Comme nous l'avons cité dans l'introduction de ce chapitre, la méthode d'estimation, due à Capéraà et al. (1997a), est définie à partir d'un échantillon d'une loi bivariée de valeurs extrêmes dont les marges sont connues. L'estimateur que nous proposons est une extension du précédent obtenu en remplaçant les observations par leur rang, et peut donc s'appliquer au cas où les marges de la loi bivariée sont inconnues.

Soit $(X_1,Y_1),...,(X_n,Y_n)$ un échantillon de loi bivariée de maxima, ayant comme fonction de répartition L et supposons tout d'abord que les lois marginales, F et G, soient connues. Nous savons que la fonction de répartition associée aux lois des valeurs extrêmes s'exprime sous la forme

$$M(x,y) = C\{F(x),G(y)\},$$

où

$$C(u,v) = P\{F(X) \leq u, G(Y) \leq v\} = exp[log(uv)A\{\frac{log(u)}{log(uv)}\}],$$

A satisfaisant les conditions énumérées au chapitre 2.

Posons $Z_i = \frac{log(U_i)}{log(U_i V_i)}, i = 1, ..., n$. La proposition suivante due à Khoudraji (1995) donne la forme de la fonction de répartition des Z_i.

Proposition 3.2. *Soit C une copule aux valeurs extrêmes de fonction de dépendance A. La fonction de répartition de la variable aléatoire $Z = log(U)/log(UV)$ est donnée par $H(z) = z + z(1-z)D(z)$ où $D(z) = A'(z)/A(z)$, $0 \leq z \leq 1$.*

De cette proposition, on obtient la caractérisation suivante:

$$\frac{A(t)}{A(s)} = exp\{\int_s^t \frac{H(z)-z}{z(1-z)}dz\}$$

où $0 \leq s \leq t \leq 1$. Ainsi, puisque $A(0) = A(1) = 1$, on obtient

$$A(t) = exp\{\int_0^t \frac{H(z)-z}{z(1-z)}dz\} = exp\{-\int_t^1 \frac{H(z)-z}{z(1-z)}dz\}. \tag{3.3}$$

Deux estimateurs non paramétriques possibles pour A sont obtenus en remplaçant H par H_n dans l'expression (3.3), soit

$$A_n^0(t) = exp\{\int_0^t \frac{H_n(z)-z}{z(1-z)}dz\}$$

$$\text{et} \tag{3.4}$$

$$A_n^1(t) = exp\{-\int_t^1 \frac{H_n(z)-z}{z(1-z)}dz\},$$

où H_n est la fonction de répartition empirique des pseudo-observations Z_i, $i = 1, ..., n$.

Capéraà et al. (1997a) ont montré que la variance asymptotique de $log\{A_n^0(t)\}$ était une fonction croissante de t et qu'a contrario celle de $log\{A_n^1(t)\}$ était une fonction décroissante. Ainsi, puisque l'estimateur $log\{A_n^0(t)\}$ apparait comme un estimateur de moins en moins fiable de $log\{A(t)\}$ lorsque t tend vers 1, et que $log\{A_n^1(t)\}$ devient également de moins en moins fiable lorsque t tend vers 0, il est alors raisonnable de considérer une combinaison de ces derniers comme estimateur de $log(A)$, attribuant plus de poids à $log\{A_n^0(t)\}$ au voisinage de 0 et à $log\{A_n^1(t)\}$ à proximité de 1. Aussi ces auteurs ont-ils suggéré comme estimateur de $log(A)$ la combinaison suivante:

$$log\{A_n(t)\} = p(t)log\{A_n^0(t)\} + \{1-p(t)\}log\{A_n^1(t)\},$$

où p est une fonction de poids bornée sur $[0,1]$ prenant la valeur 1 en 0 et la valeur 0 en 1. Ainsi $A_n(t)$ prendra la valeur 1 pour $t=0$ et $t=1$, ce qui est une propriété de la fonction de dépendance A. Dans la suite, nous prendrons $p(t)=1-t$, $0\leq t\leq 1$.

Cet estimateur de $log(A)$ est sans biais et converge presque sûrement uniformément. Dans le cas où les pseudo-observations sont toutes distinctes, la proposition suivante (Capéraà et al., 1997a) donne une expression explicite de cet estimateur.

Proposition 3.3. *Soient $Z_{(1)},...,Z_{(n)}$ les statistiques d'ordre associées aux pseudo-observations Z_i, i=1,...,n, supposées distinctes, et soit*

$$Q_i=\{\Pi_{k=1}^{i}Z_{(k)}/(1-Z_{(k)})\}^{1/n}, 1\leq i\leq n.$$

Soit p une application bornée, définie sur $[0,1]$; l'estimateur A_n défini par

$$log\{A_n(t)\}=p(t)log\{A_n^0(t)\}+\{1-p(t)\}log\{A_n^1(t)\}$$

s'écrit de façon explicite sous la forme:

$$A_n(t)=\begin{cases}(1-t)Q_n^{1-p(t)} & si\ 0\leq t\leq Z_{(1)}\\ \\ t^{i/n}(1-t)^{1-i/n}Q_n^{1-p(t)}Q_i^{-1} & si\ Z_{(i)}<t\leq Z_{(i+1)}, i=1,...,n-1\\ \\ tQ_n^{-p(t)} & si\ Z_{(n)}<t\leq 1\,.\end{cases}$$

Notons que ce nouvel estimateur, A_n, n'est pas sans biais pour A puisque, en vertu de l'inégalité de Jensen, $E(A_n)>exp[E\{log(A_n)\}]=A$. Mais A_n est un estimateur asymptotiquement sans biais pour A, fortement uniformément convergent, et ce pour une fonction de poids p bornée sur $[0,1]$.

Lorsque les lois marginales ne sont pas connues, ce qui est très souvent le cas, on peut les estimer par les lois marginales empiriques. Les pseudo-observations deviennent alors

$$Z_{in}^{*}=\frac{log\{F_n(X_i)\}}{log\{F_n(X_i)G_n(Y_i)\}}\approx\frac{log(\frac{i}{n+1})}{log\{\frac{iS_i}{(n+1)^2}\}},$$

où S_i représente le rang de Y_i lorsque les X_i sont ordonnées par ordre croissant. L'estimateur pour $log(A)$ que nous proposons s'écrit alors

$$log\{A_n^*(t)\} = p(t)log\{A_n^{0*}(t)\} + \{1 - p(t)\}log\{A_n^{1*}(t)\}$$

où $log\{A_n^{0*}(t)\}$ et $log\{A_n^{1*}(t)\}$ sont obtenus en remplaçant H_n par H_n^* dans l'expression (3.4), H_n^* étant la fonction de répartition empirique des Z_{in}^*. Cet estimateur est fortement uniformément convergent comme nous le montrons dans la proposition suivante.

Proposition 3.4. *La quantité $log(A_n^*)$ est un estimateur fortement uniformément convergent, c'est-à-dire,*

$$sup_{t\in[0,1]} \mid log\{A_n^*(t)\} - log\{A(t)\} \mid \longrightarrow 0 \ p.s., \ si \ n \to \infty.$$

Démonstration. Considérons (X, Y) suivant la loi $M(x, y)$ de copule C, c'est-à-dire que

$$M(x, y) = C\{F(x), G(y)\},$$

et $(X_1, Y_1), ..., (X_n, Y_n)$ un échantillon issu de M. Soient F_n, G_n, les fonctions de répartition marginales empiriques renormalisées,

$$F_n(x) = \frac{1}{n+1}\sum_{i=1}^{n} I_{\{X_i \leq x\}} \text{ et } G_n(y) = \frac{1}{n+1}\sum_{i=1}^{n} I_{\{Y_i \leq y\}}.$$

Soit C_n la copule empirique "renormalisée", c'est-à-dire

$$C_n(u, v) = \frac{1}{n}\sum_{k=1}^{n} I_{\{F_n(X_k)\leq u\}} I_{\{G_n(Y_k)\leq v\}}.$$

Considérons également les variables aléatoires discrètes $Z_i^* = \frac{log\{F_n(X_i)\}}{log\{F_n(X_i)G_n(Y_i)\}}$, $i = 1, ..., n$, possédant n^2 valeurs possibles, $z_{n,1,n} \leq ... \leq z_{1,n,n}$, où

$$z_{n,1,n} = \frac{1}{1 + \frac{log(\frac{1}{n+1})}{log(\frac{n}{n+1})}} = \frac{log(\frac{n}{n+1})}{log\{\frac{n}{(n+1)^2}\}}.$$

Soit H_n^* la fonction de répartition empirique associée à ces Z_i^*,

$$H_n^*(x) = \frac{1}{n}\sum_{i=1}^{n} I_{\{Z_i^* \leq z\}}$$

et soit H la fonction de répartition de la variable aléatoire $Z = \frac{log\{F(x)\}}{log\{F(x)G(y)\}}$. On doit montrer que

$$|log\{A_n^{0*}(t)\} - log\{A(t)\}| \longrightarrow 0 \text{ p.s., si } n \to \infty, \ t \in [0,1],$$

où $log\{A_n^{0*}(t)\} = \int_0^t \frac{H_n^*(z)-z}{z(1-z)}dz$ et $log\{A(t)\} = \int_0^t \frac{H(z)-z}{z(1-z)}dz$.

1) Considérons le cas où $t \in (0, 1/2]$.

On a

$$\begin{aligned} |log\{A_n^{0*}(t)\} - log\{A(t)\}| &= |\int_0^t \frac{(H_n^* - H)(z)}{z(1-z)}dz| \\ &\leq \int_0^{z_{n,1,n}} \frac{H(z)}{z(1-z)}dz + \frac{1}{1-t} sup_{[z_{n,1,n},t]}|H_n^* - H|(z) \int_{z_{n,1,n}}^t \frac{dz}{z} \end{aligned}$$

car $z_{n,1,n} \to 0$, donc à partir d'un certain rang on a $z_{n,1,n} \leq t$. Comme $H(z)/\{z(1-z)\}$ est une quantité bornée (voir démonstration de Proposition 3.1, Capéraà et al., 1997a), on peut écrire, $\forall \epsilon > 0$ et à partir d'un certain rang n_ϵ,

$$|log\{A_n^{0*}(t)\} - log\{A(t)\}| \leq \epsilon + \frac{1}{1-t} sup_{[z_{n,1,n},t]}|H_n^* - H|(z)|log(z_{n,1,n})|,$$

et puisque $t \in [0, 1/2]$

$$sup_{[0,1/2]}|log\{A_n^{0*}(t)\} - log\{A(t)\}| \leq \epsilon + 2 sup_{[z_{n,1,n},1/2]}|H_n^* - H|(z)|log(z_{n,1,n})|.$$

Premièrement, montrons que $\forall \alpha < 1/2$,

$$n^\alpha sup_{[0,1/2]}|H_n^* - H|(z) \longrightarrow 0 \text{ p.s., si } n \to \infty.$$

Exprimons tout d'abord H_n^* et H en terme de C_n et C.

$$\begin{aligned} H_n^*(z) &= \frac{1}{n}\sum_{i=1}^n I[\frac{log\{F_n(X_i)\}}{log\{F_n(X_i)G_n(Y_i)\}} \leq z] \\ &= \int\int_{[0,1]^2} I_{[0,z]}\{\frac{log(u)}{log(uv)}\}dC_n(u,v), \end{aligned}$$

et de même on a

$$H(z) = \int\int_{[0,1]^2} I_{[0,z]}\{\frac{log(u)}{log(uv)}\}dC(u,v).$$

Ainsi,

$$\begin{aligned}(H_n^* - H)(z) &= \int\int_{[0,1]^2} I_{[0,z]}\{\frac{log(u)}{log(uv)}\}d(C_n - C)(u,v) \\ &= \int\int I_{D_z}(u,v)d(C_n - C)(u,v)\end{aligned}$$

où $D_z = \{(u,v) \in (0,1)^2 : v \leq u^{(\frac{1-z}{z})}\}$.

Faisons une intégration par parties dont nous rappelons les conditions (Hildebrandt, 1963, Théorème 8.8, p.127).

Théorème. *Soient 2 fonctions f et g de $[a,a']$ x $[b,b']$ dans R telles que*

$$\int_{[a,a'] \times [b,b']} \{f(x,y) - f(x,b) - f(a,y) + f(a,b)\}dg(x,y)$$

existe, alors

$$\int_{[a,a'] \times [b,b']} \{g(x,y) - g(x,b') - g(a',y) + g(a',b')\}df(x,y)$$

existe, et ces deux intégrales sont égales.

En posant $g_z(x,y) = I_{D_z}(x,y)$ et $f_n(x,y) = n^\alpha(C_n - C)(x,y)$ sur $[0,1]$ x $[0,1]$, et en appliquant le résultat de ce théorème, puisque $g_z(x,1) = f_n(0,y) = f_n(x,0) = f_n(0,0) = 0$ et $g_z(1,y) = g_z(1,1) = 1$, on obtient

$$\begin{aligned}sup_{z\in[0,1]}|\int\int_{[0,1]^2} f_n(x,y)dg_z(x,y)| &\leq sup_{[0,1]^2}|f_n(x,y)|\ V_g \\ &= n^\alpha sup_{[0,1]^2}|C_n - C|(x,y)\end{aligned}$$

où $V_g = 1$ représente la variation totale de g. D'après le Théorème 3.1 de Deheuvels (1979), on a

$$\forall\alpha < 1/2,\ n^\alpha sup_{[0,1]^2}|C_n - C|(x,y) \to 0\ p.s.$$

On en déduit d'après le Théorème 8.8 de Hildebrandt cité ci-haut que

$$n^\alpha sup_{[0,1]}|H_n^* - H|(z) = sup_{z\in[0,1]}|\int\int_{[0,1]^2} g_z(x,y)df_n(x,y)|$$

converge presque sûrement vers zéro uniformément.

Montrons maintenant que $\frac{|log(z_{n,1,n})|}{n^\alpha}$ est une quantité bornée. En se servant de l'expression de $z_{n,1,n}$,

$$z_{n,1,n} = \frac{log(\frac{n}{n+1})}{log\{\frac{n}{(n+1)^2}\}}$$

et en posant $x = \frac{1}{n+1}$, on obtient, pour n assez grand

$$\begin{aligned}\frac{1}{n^\alpha} log(z_{n,1,n}) &= \frac{1}{n^\alpha} log[\frac{log(\frac{n}{n+1})}{log\{\frac{n}{(n+1)^2}\}}] \\ &\approx x^\alpha log[\frac{log(1-x)}{log\{x(1-x)\}}] \\ &\approx x^\alpha [log\{-log(1-x)\} - log[-log\{x(1-x)\}]] \\ &\approx x^\alpha log(x) - x^\alpha log\{-log(x)\}.\end{aligned}$$

Or, lorsque $x \to 0$,

$$x^\alpha log(x) \to 0$$

et en posant $u = -log(x)$, on obtient si $x \to 0$

$$x^\alpha log\{-log(x)\} = e^{-\alpha u} log(u) \to 0.$$

On en déduit que pour $\alpha < 1/2$

$$\frac{|log(z_{n,1,n})|}{n^\alpha} \to 0, \ si\ n \to \infty.$$

Puisque

$$n^\alpha sup_{[0,1]} |H_n^* - H|(z) \longrightarrow 0 \text{ p.s., si } n \to \infty,$$

et que

$$\frac{|log(z_{n,1,n})|}{n^\alpha} \to 0, \ si\ n \to \infty,$$

alors

$$2sup_{[z_{n,1,n},1/2]} |H_n^* - H|(z) |log(z_{n,1,n})| \longrightarrow 0 \text{ p.s., si } n \to \infty,$$

d'où

$$sup_{[0,1/2]} |log\{A_n^{0*}(t)\} - log\{A(t)\}| \longrightarrow 0 \text{ p.s., si } n \to \infty.$$

2) Soit maintenant $t \in (1/2, 1]$.

On a

$$|log\{A_n^{0*}(t)\} - log\{A(t)\}| \leq sup_{[0,1/2]} |log\{A_n^{0*}(t)\} - log\{A(t)\}| + |\int_{1/2}^t \frac{(H_n^* - H)(z)}{z(1-z)} dz|.$$

De plus,

$$\begin{aligned}|\int_{1/2}^t \frac{(H_n^* - H)(z)}{z(1-z)} dz| &\leq 2sup_{[1/2,t]} |H_n^* - H|(z) |\int_{1/2}^t \frac{dz}{1-z}| \\ &\leq 2sup_{[1/2,t]} |H_n^* - H|(z) log(\frac{1}{1-t})\end{aligned}$$

Par ailleurs,

$$z_{1,n,n} = \frac{1}{1 + \frac{log\{n/(n+1)\}}{log\{1/(n+1)\}}} = \frac{log(n+1)}{log\{\frac{(n+1)^2}{n}\}} \to 1, \;\; si \;\; n \to \infty,$$

donc $t < z_{1,n,n}$, à partir d'un certain rang.

Par suite,

$$|\int_{1/2}^{t} \frac{(H_n^* - H)(z)}{z(1-z)} dz| \leq 2sup_{[1/2,t]}|H_n^* - H|(z)log(\frac{1}{1 - z_{1,n,n}}).$$

En faisant une intégration par parties comme précédemment, on obtient

$$n^{\alpha} sup_{[1/2,t]}|H_n^* - H|(z) \longrightarrow 0 \text{ p.s., si } n \to \infty.$$

De plus, on peut vérifier que

$$\frac{1}{1 - z_{1,n,n}} = \frac{1}{z_{n,1,n}}$$

donc

$$log(\frac{1}{1 - z_{1,n,n}}) = log(z_{n,1,n}),$$

et puisque $\frac{|log(z_{n,1,n})|}{n^{\alpha}} \to 0$, on conclut:

$$sup_{[1/2,1]}|log\{A_n^{0*}(t)\} - log\{A(t)\}| \longrightarrow 0 \text{ p.s., si } n \to \infty.$$

De la même façon, on peut montrer que

$$sup_{[1/2,1]}|log\{A_n^{1*}(t)\} - log\{A(t)\}| \longrightarrow 0 \text{ p.s., si } n \to \infty,$$

ce qui complète la preuve.

3.2 Méthode du seuil de Khoudraji

Cette méthode d'estimation de la fonction de dépendance A a été esquissée par de Haan (1985). Supposons que (X_i, Y_i), $i = 1, ..., n$, suit une loi F dont les marginales F_1 et F_2 sont des lois de Fréchet unités. Considérons les variables $R_i = X_i + Y_i$ et $W_i = X_i/R_i$, $i = 1, ..., n$, alors, d'après la relation (2.6), la fonction A est liée à la fonction S par

$$A(t) = \int_0^1 max\{wt, (1-w)(1-t)\}dS(w). \tag{3.5}$$

De plus, d'après la Proposition 3.1, nous avons

$$\lim_{u\to\infty} P\left(\frac{X}{X+Y} \leq w \mid X+Y > u\right) = \frac{S(w)}{2}, \tag{3.6}$$

d'où l'on déduit que $S/2$ est la fonction de répartition conditionnelle de W_i, sachant que R_i est supérieur à u, et ce lorsque u tend vers l'infini. On peut alors estimer A en remplaçant dans (3.5) $S/2$ par $S_n/2$, la fonction de répartition empirique des W_i pour lesquels les R_i sont supérieurs à u. Le choix de u doit être fait en tenant compte, entre autres, de l'équation (3.6) qui implique que u doit être assez grand pour que les W_i et les R_i, tels que $R_i > u$, soient indépendants. Cependant, pour une taille d'échantillon fixée, on ne peut pas choisir u trop grand car on risque de ne pas avoir alors assez d'observations pour estimer convenablement A. Aussi cette valeur $u = u(n)$ dépendra de la taille n de l'échantillon et nous appellerons seuil de l'estimateur de Khoudraji la quantité $u(n)/n$. Dans les chapitres qui vont suivre nous considèrerons plusieurs seuils possibles.

Nous avons vu plus haut que, dans cette approche, on faisait l'hypothèse que les lois marginales de la loi F des (X_i, Y_i) étaient des lois de Fréchet. Comme cette condition n'est pas en général vérifiée, on peut la satisfaire en effectuant le changement de variables suivant:

$$X_i^* = -\frac{1}{log\{F_1(X_i)\}} \text{ et } Y_i^* = -\frac{1}{log\{F_2(Y_i)\}},\ i = 1, ..., n.$$

En effet on a, par exemple, pour X_i^*,

$$P(X_i^* \leq x) = P[log\{F_1(X_i)\} \leq -\frac{1}{x}] = P\{F_1(X_i) \leq e^{-1/x}\} = e^{-1/x},\ x \geq 0.$$

Comme les lois F_1 et F_2 ne sont pas connues, on peut les remplacer par les fonctions de répartition empiriques correspondantes. Ainsi, si n n'est pas trop petit, les nouvelles variables X_i^* et Y_i^* définies par

$$X_i^* = -\frac{1}{log(\frac{Q_{xi}-1/2}{n})}$$

et

$$Y_i^* = -\frac{1}{log(\frac{Q_{yi}-1/2}{n})}$$

où Q_{xi} et Q_{yi} sont les rangs de X_i et Y_i respectivement, suivent approximativement des lois de Fréchet unités. Considérons alors les variables

$$R_i = X_i^* + Y_i^*$$

et

$$W_i = X_i^*/R_i,$$

on obtient l'estimateur de A suivant:

$$\hat{A}_n(t) = \int max\{wt, (1-w)(1-t)\} dS_n(w) = \frac{2}{n^*} \sum_{i=1}^{n^*} max\{W_i t, (1-W_i)(1-t)\},$$

où n^* représente le nombre de W_i correspondant aux $R_i > u(n)$, et ce pour un choix de $u(n)$ assez grand pour que les W_i et les R_i soient indépendants.

3.3 Méthode d'estimation de Joe, Smith et Weissman

Supposons toujours que la loi F de (X_i, Y_i), $i = 1, ..., n$, a pour marginales des lois de Fréchet unités. Le processus ponctuel associé à $\{T_i(\beta) = n^{-1} max(X_i, \beta^{-1} Y_i); i = 1, ..., n\}$ tend vers un processus de Poisson non-homogène défini sur $(0, +\infty)$ et d'intensité μ vérifiant, pour $y > 0$ et $\beta > 0$,

$$\mu(y, \beta y) = y^{-1} \mu(1, \beta).$$

Ce résultat asymptotique ne peut s'appliquer que pour n assez grand et pour les $T_i(\beta)$ assez grands eux-aussi, c'est-à-dire supérieurs à un certain seuil T. Considérons les $T_i(\beta)$ supérieurs ou égaux à T. On déduit de ce résultat asymptotique que l'espérance de leur nombre est approximativement égal à $T^{-1}\mu(1, \beta)$. Si l'on prend maintenant $T = T_{[k]}(\beta)$, où $T_{[k]}(\beta)$ est le kième plus grand $T_i(\beta)$, alors un estimateur de $\mu(1, \beta)$ est $kT_{[k]}(\beta)$ puisqu'il y a k pseudo-observations $T_i(\beta)$ plus grandes que $T_{[k]}(\beta)$. Puisque ce raisonnement peut-être fait pour tous les $T_i(\beta)$ plus grands ou égaux au seuil T, que l'on déterminera plus loin, on peut remplacer $kT_{[k]}(\beta)$ par une moyenne des $iT_{[i]}(\beta)$ à condition que ces derniers soient approximativement constants. Ceci devrait permettre de diminuer la variabilité de $\mu(1, \beta)$. D'après la relation (3.2), on a

$$\mu(1, \beta) = (1 + \beta^{-1}) A(\frac{\beta}{1+\beta}),$$

et un estimateur de A sera alors,

$$\hat{A}(\frac{\beta}{1+\beta}) = \sum_i \frac{iT_{[i]}(\beta)}{\hat{n}} \frac{1}{1+\beta^{-1}}$$

où $\tilde{n}$ est le nombre de $T_i(\beta)$ pour lesquels les $iT_{[i]}(\beta)$ sont approximativement constants.

Afin de vérifier que n et T sont assez grands pour pouvoir appliquer le résultat asymptotique, on peut s'appuyer sur la Proposition 5.17 de Resnick (1987) qui établit la relation suivante entre l'intensité μ du processus asymptotique et la loi F des (X_i, Y_i), $i = 1, ..., n$,

$$\lim_{t\to\infty} \frac{1 - F(tx, ty)}{1 - F(t, t)} = \frac{\mu(x, y)}{\mu(1, 1)}.$$

Considérons maintenant pour t assez grand, $y \geq t$ et en posant $x = y/t$,

$$\begin{aligned} P\{max(X_i, \beta^{-1}Y_i) \geq y | max(X_i, \beta^{-1}Y_i) \geq t\} &= \frac{1 - F(y, \beta y)}{1 - F(t, \beta t)} \\ &= \frac{1 - F(tx, \beta tx)}{1 - F(t, \beta t)} \\ &\approx \frac{\mu(x, \beta x)}{\mu(1, \beta)} \\ &= x^{-1} \\ &= \frac{t}{y}. \end{aligned}$$

Ainsi, lorsque T est assez grand et β fixé, on peut considérer que les $T_i(\beta)$ supérieurs ou égaux à T suivent, conditionnellement, la loi de densité $\frac{T}{y^2}$, $y \geq T$, qui est telle que $E(iT_{[i]}(\beta))$ est approximativement indépendante de i pour i pas trop voisin de 1. En effet, en supposant que $T_1(\beta), ..., T_{\tilde{n}}(\beta)$ est un échantillon de la loi de densité $\frac{T}{y^2}$, $y \geq t$, $\tilde{n}$ étant le nombre de $T_i(\beta)$ supérieurs ou égaux à T, on obtient $E(iT_{[i]}(\beta)) = \frac{i}{i-1}\tilde{n}T$. On peut donc choisir T de telle façon que les $iT_{[i]}(\beta)$ soient approximativement constants. C'est la solution suggérée par Joe et al. Afin de respecter l'hypothèse que les lois marginales de F sont des lois de Fréchet unités, on remplace les variables X_i et Y_i par X_i^* et Y_i^* définies à la Section 3.2.

En ce qui concerne les deux estimateurs proposés respectivement par Khoudraji et Joe et al., nous ne possédons malheureusement pas de propriétés asymptotiques, ni même pour de petits échantillons. Dans le chapitre suivant, nous allons essayer de combler ce manque à partir de simulations.

Chapitre 4

Comparaison des estimateurs

L'objectif principal de ce chapitre est de comparer au moyen de simulations, et ce pour de petits échantillons, les propriétés des estimateurs de la fonction de dépendance A^* présentées au chapitre précédent. Cette comparaison va porter essentiellement sur le biais possible des estimateurs que l'on va mesurer de deux façons différentes. Ayant fixé la fonction de dépendance A^* d'une copule extrême, nous allons déterminer la force de sa dépendance en calculant le tau de Kendall qui est donné par (Khoudraji, 1995)

$$\tau_{A^*} = \int_0^1 \frac{t(1-t)}{A^*(t)} dA'^*(t).$$

Pour chaque estimateur, nous calculerons premièrement les tau de Kendall obtenus à partir des estimations de A^* découlant des différents échantillons simulés, et deuxièmement, les distances L_1 entre A^* et ses estimations. Ainsi, ayant pour chaque échantillon simulé une estimation du tau de Kendall de A^* et une valeur de la distance L_1, nous comparerons les estimateurs selon ces deux mesures ainsi que leur variabilité au moyen de diagrammes en boîte (box-plot).

Les simulations seront faites à partir de lois Archimax, définies par Capéraà et al. (1997b). Ces lois, appartenant au domaine d'attraction de différentes copules de valeurs extrêmes, seront choisies avec des forces de dépendance différentes. De plus, la fonction de dépendance qui leur est associée sera symétrique ou asymétrique.

Dans la Section 4.1, nous rappelons la définition des copules Archimax ainsi que leurs algorithmes de simulation. La section suivante est consacrée à la présentation des estimateurs retenus dans cette étude. Dans la Section 4.3, un plan factoriel est proposé pour générer les données provenant des simulations; des analyses de variance suivront pour étayer les comparaisons. Une conclusion générale termine ce chapitre.

4.1 Copules Archimax

Une copule bivariée est appelée copule Archimax si et seulement si elle peut s'écrire sous la forme

$$C_{\phi,A}(x,y) = \phi^{-1}[\{\phi(x)+\phi(y)\}A\{\frac{\phi(x)}{\phi(x)+\phi(y)}\}],$$

pour tout $0 \leq x, y \leq 1$, en terme d'une fonction convexe décroissante ϕ: $(0,1] \to [0,\infty)$, vérifiant $\phi(1) = 0$, et d'une fonction convexe A: $[0,1] \to [1/2,1]$, telle que $max(t, 1-t) \leq A(t) \leq 1$, pour tout $0 \leq t \leq 1$, avec la convention que $\phi^{-1}(t) = 0$ lorsque $t \geq \lim_{s\to 0}\phi(s)$. La fonction ϕ est appelée générateur de la copule.

Exemple 4.1: *Soit la fonction de dépendance du modèle mixte due à Tawn (1988) donnée par*

$$A(t) = \theta t^2 - \theta t + 1,\ 0 \leq t \leq 1,\ 0 \leq \theta \leq 1,$$

et considérons le générateur de Clayton

$$\phi_{1,\alpha}(t) = \frac{1}{\alpha}(t^{-\alpha}-1),\ \alpha \geq 0,$$

et celui de Franck

$$\phi_{2,\alpha}(t) = -log(\frac{1-e^{-t\alpha}}{1-e^{-\alpha}}),\ \alpha \in R,$$

alors $C_{\phi_{1,\alpha},A}$ et $C_{\phi_{2,\alpha},A}$ sont des lois Archimax. Ces lois seront parmi celles utilisées lors des simulations présentées à la Section 4.3.

Il est important de remarquer que l'ensemble des copules Archimax $C_{\phi,A}$ contient les copules de valeurs extrêmes ainsi que les copules archimédiennes. En effet, si l'on pose $\phi(t) = log(1/t)$, la copule $C_{\phi,A}$ est alors une copule de valeurs extrêmes, c'est-à-dire

que
$$C_{\phi,A} = C_A(x,y) = exp[\{log(x) + log(y)\}A\{\frac{log(x)}{log(x) + log(y)}\}].$$
De même, si la fonction $A \equiv 1$ alors, on obtient une copule archimédienne:
$$C_{\phi,A} = C_\phi(x,y) = \phi^{-1}\{\phi(x) + \phi(y)\}.$$

La proposition suivante (Capéraà et al., 1997b), établit une relation entre la fonction de dépendance de la copule Archimax $C_{\phi,A}$ et la fonction de dépendance de son attracteur.

Proposition 4.1. *Soit $C_{\phi,A}$ une copule bivariée de fonction de dépendance A et de générateur ϕ tel que $\phi(1-1/t)$ est à variations régulières de degré $-m$, $m \geq 1$. Alors $C_{\phi,A}$ est située dans le domaine d'attraction d'une loi de valeurs extrêmes ayant comme fonction de dépendance*
$$A^*(t) = \{t^m + (1-t)^m\}^{1/m} A^{1/m}\{\frac{t^m}{t^m + (1-t)^m}\},\ 0 \leq t \leq 1. \tag{4.1}$$

L'équation (4.1) nous permet d'obtenir A en fonction de A^* et de m, soit
$$A(t) = \{t^{1/m} + (1-t)^{1/m}\}^m [A^*\{\frac{t^{1/m}}{t^{1/m} + (1-t)^{1/m}}\}]^m.$$
Ainsi, pour construire une famille paramétrique de copule Archimax bivariée ayant comme attracteur maximum C_{A^*}, il suffit de choisir une famille de copule archimédienne ayant un générateur ϕ tel que $\phi(1-1/t)$ est à variations régulières de degré $-m$, et de choisir $m \geq 1$ tel que A soit une fonction de dépendance. On montre que cette dernière condition est satisfaite si et seulement si il existe $l \geq m$ tel que $A^* \leq G_l$, où
$$G_l(t) = \{t^l + (1-t)^l\}^{1/l},\ 0 \leq t \leq 1$$
est la fonction de dépendance d'une copule de valeurs extrêmes de Gumbel, de paramètre $l \geq 1$ (voir justification dans Capéraà et al., 1997b). Notons que $l = 1$ peut toujours être considéré comme une possibilité pour le choix de l mais, puisque $m \geq 1$, on en déduit alors que $m = 1$ et donc $A = A^*$. C'est le cas notamment des générateurs de Clayton et de Franck cités dans l'Exemple 4.1 pour lesquels on a $m = 1$.

Pour générer des observations d'une copule Archimax $C_{\phi,A}(x,y)$, deux méthodes simples nous sont proposées. La première suggère de générer X de loi uniforme $[0,1]$ et de générer Y de loi conditionnelle $\partial C_{\phi,A}(x,y)/\partial x$. Mais puisque cette dernière n'est généralement pas inversible, il sera nécessaire d'utiliser un deuxième algorithme dont la justification est donnée dans Capéraà et al. (1997b):

i) Générer z de fonction de répartition: $H(z) = z + (1-z)\frac{A'(z)}{A(z)}$, $0 \leq z \leq 1$;

ii) Etant donné z, générer u de loi uniforme sur l'intervalle $[0,1]$;

iii) Si $u \leq p(z) = \frac{z(1-z)A''(z)}{h(z)A(z)}$, générer w de loi uniforme sur $[0,1]$; sinon générer w de loi $K_\phi(t) = t - \lambda_\phi(t)$ où $\lambda_\phi(t) = \phi(t)/\phi'(t)$;

iv) Déterminer x et y par $x = \phi^{-1}\{z\phi(w)/A(z)\}$ et $y = \phi^{-1}\{(1-z)\phi(w)/A(z)\}$.

Lorsque la fonction de répartition H n'est pas inversible, on utilise alors pour déterminer z la méthode d'acceptation-rejet, la densité de H étant donnée par

$$h(z) = 1 + (1-2z)A'(z)/A(z) + z(1-z)[A''(z)/A(z) - \{A'(z)/A(z)\}^2].$$

4.2 Choix des estimateurs

4.2.1 Estimateur de Capéraà, Fougères et Genest

Nous avons vu au chapitre précédent que l'estimation de la fonction de dépendance A^* sera considérée dans les deux situations suivantes. Dans la première, l'échantillon dont nous disposons est de loi F bidimensionnelle, c'est-à-dire que les observations ne sont pas des maxima. Dans la deuxième, l'estimation est faite à partir d'un échantillon d'une loi bivariée de valeurs extrêmes.

La méthode de Capéraà et al. s'appliquent à des lois de valeurs extrêmes. Si l'échantillon simulé ne provient pas de telles lois, il est nécessaire de générer, à partir de celui-ci, un échantillon d'observations pouvant être considéré comme extrêmes. Pour ce faire, nous utiliserons la méthode dite des "paquets", utilisée par plusieurs auteurs comme, par exemple, Pickands III (1975). Cette méthode consiste en la formation de sous-ensembles (paquets) de couples (X_i, Y_i) de loi F, tous de même taille, et pour lesquels le maximum sur chacune des deux coordonnées sera seul retenu. Si par exemple, nous avons un échantillon de 2500 couples (X_i, Y_i), on peut choisir de séparer les 2500 couples en 25 paquets de 100 couples (X_i, Y_i), c'est-à-dire que le premier paquet sera formé des couples (X_i, Y_i) pour $i = 1, ..., 100$, le 2e paquet, des couples (X_i, Y_i) pour $i = 101, ..., 200$, et ainsi de suite. Sur chacun des 25 paquets, le maximum sur les X_i et le maximum sur les Y_i sont retenus, ce qui formera le couple $(maxX_i, maxY_i)$. Nous aurons ainsi au total 25 couples de maxima. C'est ce nouvel échantillon qui sera alors utilisé pour calculer l'estimateur de Capéraà et al. dénoté dans la suite par CFG.

4.2.2 Méthode de Khoudraji

Nous avons vu à la Section 3.2 que, pour que l'estimateur proposé par Khoudraji soit valable, il est nécessaire de choisir un seuil r tel que les W_i et les R_i, où $R_i > nr$, soient indépendants, et tel que le nombre d'observations pour estimer convenablement A soit assez grand. Lors des simulations, nous avons remarqué que le nombre n^* d'observations (X_i, Y_i) pour lesquelles les R_i/n correspondants sont supérieurs à un seuil r était approximativement égal à $\frac{2}{r}$. Ceci peut s'expliquer de la manière suivante. Le processus ponctuel associé à $\{\frac{(X_i,Y_i)}{n}, i = 1, ..., n\}$ tend vers un processus de Poisson non homogène sur $R^{+2}\backslash\{0,0\}$ dont l'intensité μ est telle que, pour $a > 0$, (voir la Proposition 5.11 de Resnick, 1987)

$$\mu\{(x,y) \in R^{+2} : x + y > a\} = 2/a.$$

On en déduit alors que, pour n assez grand et r pas trop proche de 0, le nombre n^* de R_i/n supérieurs à r est approximativement égal à $\frac{2}{r}$.

Pour les simulations que nous avons effectuées, plusieurs niveaux de seuil ont été choisis. Pour la première analyse, ayant simulé des échantillons de taille 2500 de copules Archimax, nous avons choisi les seuils 0.01, 0.04 et 0.08. Pour ces trois seuils, le nombre d'observations pour estimer A^* devrait être approximativement de 200, 50 et 25 respectivement. Nous aurons donc 3 estimateurs différents dénotés K_1, K_2 et K_3 correspondant au seuil 0.01, 0.04 et 0.08 respectivement. Dans la deuxième analyse, portant sur des échantillons de tailles 50 et 100 issus de lois de valeurs extrêmes, nous avons choisi pour chacune de ces tailles les seuils 0.1 et 0.2. Ainsi, pour estimer A^*, on aura approximativement 10 et 20 observations pour les échantillons de taille 50, et 20 et 40 pour les échantillons de taille 100. Nous notons K_1^* l'estimateur correspondant au seuil 0.1 et K_2^* celui correspondant au seuil 0.2.

Enfin nous avons vérifié dans les deux situations, et sur quelques échantillons, l'indépendance entre les W_i et les R_i en calculant le tau de Kendall. Par exemple, pour un échantillon particulier de taille 2500, nous avons obtenu les tau de Kendall suivants: -0.00954 (seuil de 0.01), -0.00421 (seuil de 0.04) et 0.00167 (seuil de 0.08), ce qui nous permet d'accepter l'hypothèse d'indépendance entre les W_i et les R_i.

4.2.3 Méthode de Joe, Smith et Weissman

Dans la méthode de Joe et al. présentée à la Section 3.3, l'estimation de la fonction de dépendance A^* repose sur une moyenne des $iT_{[i]}(\beta)$ pour lesquels les $T_i(\beta)$ sont supérieurs à un certain seuil T. De plus, pour que cette méthode soit applicable, il est nécessaire que les valeurs $iT_{[i]}(\beta)$ utilisées pour calculer cette moyenne soient approximativement constantes. Dans la première analyse portant sur des échantillons de taille 2500, plutôt que de choisir 3 seuils différents, nous avons décidé de conserver les 200, 50 et 25 plus grandes pseudo-observations $T_i(\beta)$, quel que soit $\beta > 0$. Ensuite, nous avons vérifié dans quelques cas, quelles étaient les valeurs $iT_{[i]}(\beta)$ constantes. Par exemple, pour un échantillon particulier et pour $\beta = 99$, lorsque 200 pseudo-observations $T_i(99)$ sont retenues, on vérifie aisément sur le graphique suivant que les 15 dernières valeurs $iT_{[i]}(99)$ ne sont pas constantes.

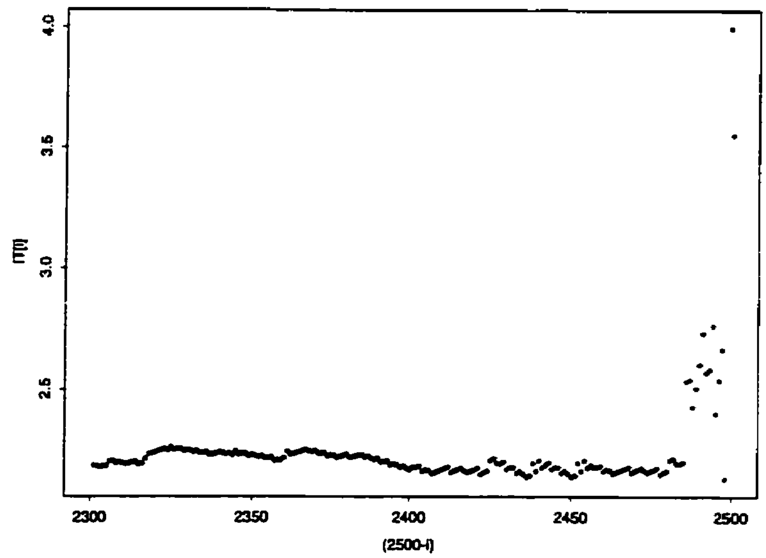


Figure 4.1: *Graphique des 200 valeurs $iT_{[i]}(99)$ pour un certain échantillon.*

Ainsi, les trois estimateurs choisis dans cette première analyse seront obtenus avec les moyennes tronquées suivantes $\frac{1}{185}\sum_{i=16}^{200} iT_{[i]}(\beta)$, $\frac{1}{35}\sum_{i=16}^{50} iT_{[i]}(\beta)$ et $\frac{1}{10}\sum_{i=16}^{25} iT_{[i]}(\beta)$. Ils seront dénotés J_1, J_2 et J_3 respectivement. On peut remarquer que nous ne faisons pas dépendre de β le nombre de $iT_{[i]}(\beta)$ composant ces différentes moyennes tronquées, ce qui est la solution proposée par Joe et al. que nous adoptons ici.

Dans la deuxième analyse portant sur des échantillons de taille 50 et 100 issus de lois de valeurs extrêmes, nous avons décidé de retenir les 20 et 10 plus grandes pseudo-observations $T_i(\beta)$. En agissant comme précédemment nous choisissons les deux estimateurs obtenus avec les moyennes tronquées $\frac{1}{15}\sum_{i=6}^{20} iT_{[i]}(\beta)$ et $\frac{1}{5}\sum_{i=6}^{10} iT_{[i]}(\beta)$ pour chacune des deux tailles d'échantillon. Ces estimateurs seront dénotés J_1^* et J_2^* respectivement.

4.3 Analyse de variance

Pour comparer les 3 méthodes d'estimation, soit celle de Capéraà et al., de Khoudraji, et de Joe et al., nous procèderons à l'étude de 2 analyses différentes. La première analyse utilisera des observations de copules Archimax, tandis que la deuxième utilisera des observations de copules de valeurs extrêmes.

4.3.1 Analyse 1

Dans cette analyse, les échantillons simulés proviennent de copules Archimax dont les fonctions de dépendance sont les mêmes que celles de leurs attracteurs C_{A^*}. Ces copules ont la forme suivante

$$C_{\phi,A^*}(x,y) = \phi^{-1}[\{\phi(x)+\phi(y)\}A^*\{\frac{\phi(x)}{\phi(x)+\phi(y)}\}],$$

où A^* est de l'un des 4 types suivants:

$$A_1^*(t) = \theta t^2 - \theta t + 1, \text{ avec } \theta = 0.3,$$

$$A_2^*(t) = \begin{cases} \frac{15}{8}t^2 - \frac{3}{4}t + 1 & \text{si } t \leq 1/5 \\ \\ \frac{1}{128}(15t^2 - 6t + 119) & \text{si } t > 1/5, \end{cases}$$

$$A_3^*(t) = \theta t^2 - t + 1, \text{ avec } \theta = 0.668,$$

$$A_4^*(t) = \begin{cases} \frac{3}{2}t^2 - t + 1 & \text{si } t < 1/3 \\ \\ \frac{1}{8}(3t^2 - 2t + 7) & \text{si } t \geq 1/3, \end{cases}$$

où A_1^* et A_3^* sont symétriques et A_2^* et A_4^* asymétriques. Les paramètres ont été choisis de telle façon que $\tau_{A_1^*} = \tau_{A_2^*} \approx 0.10$ et $\tau_{A_3^*} = \tau_{A_4^*} \approx 0.26$, ce qui donne deux forces de dépendance pour les attracteurs C_{A^*}.

Les générateurs ϕ sont ceux de Clayton (ϕ_1) et de Franck (ϕ_2) définis à l'Exemple 4.1. Le paramètre α a été calculé pour chacun de ces générateurs de telle façon que l'on obtienne $\tau_{\phi_i,A_1^*} = 0.30$ et $\tau_{\phi_i,A_2^*} = 0.50$, pour $i = 1, 2$, où

$$\tau_{\phi,A^*} = \tau_{A^*} + (1 - \tau_{A^*})\tau_\phi \tag{4.2}$$

et

$$\tau_\phi = 1 + 4\int_0^1 \frac{\phi}{\phi'}(t)dt.$$

Si l'on prend les mêmes valeurs de α pour les copules Archimax C_{ϕ,A_3^*} et C_{ϕ,A_4^*} que pour C_{ϕ,A_1^*} et C_{ϕ,A_2^*}, alors d'après (4.2) nous ne pouvons pas vérifier $\tau_{\phi_i,A_3^*} = 0.30$, $i = 1, 2$, et $\tau_{\phi_i,A_4^*} = 0.50$, $i = 1, 2$. Nous allons plutôt choisir ce paramètre α de façon à obtenir $\tau_{\phi_i,A_3^*} = 0.50$ et $\tau_{\phi_i,A_4^*} = 0.70$. Ceci va nous permettre de voir l'influence de la force de dépendance des copules Archimax sur les estimateurs de A^*, dans chacun des cas $\tau_{A^*} = 0.10$ et $\tau_{A^*} = 0.26$. Notons que les deux générateurs $\phi_{i,\alpha}$, $i = 1, 2$, étant tels que $\phi_{i,\alpha}(1 - 1/t)$ est à variations régulières de degré -1, les copules Archimax $C_{\phi_{i,\alpha},A^*}$ sont dans le domaine d'attraction de C_{A^*}.

Le Tableau 4.1 résume tous ces choix.

τ_{A^*}	$A^*(t)$	$\phi(t)$	τ_{ϕ,A^*}
0.1	A_1^*	$\phi_{1,\alpha}$ avec $\alpha = 4/7$	0.30
0.1	A_1^*	$\phi_{1,\alpha}$ avec $\alpha = 1.6$	0.50
0.1	A_1^*	$\phi_{2,\alpha}$ avec $\alpha = 2.09$	0.30
0.1	A_1^*	$\phi_{2,\alpha}$ avec $\alpha = 4.8$	0.50
0.1	A_2^*	$\phi_{1,\alpha}$ avec $\alpha = 4/7$	0.30
0.1	A_2^*	$\phi_{1,\alpha}$ avec $\alpha = 1.6$	0.50
0.1	A_2^*	$\phi_{2,\alpha}$ avec $\alpha = 2.09$	0.30
0.1	A_2^*	$\phi_{2,\alpha}$ avec $\alpha = 4.8$	0.50
0.26	A_3^*	$\phi_{1,\alpha}$ avec $\alpha = 0.972$	0.50
0.26	A_3^*	$\phi_{1,\alpha}$ avec $\alpha = 2.953$	0.70
0.26	A_3^*	$\phi_{2,\alpha}$ avec $\alpha = 3.23$	0.50
0.26	A_3^*	$\phi_{2,\alpha}$ avec $\alpha = 7.83$	0.70
0.26	A_4^*	$\phi_{1,\alpha}$ avec $\alpha = 0.972$	0.50
0.26	A_4^*	$\phi_{1,\alpha}$ avec $\alpha = 2.953$	0.70
0.26	A_4^*	$\phi_{2,\alpha}$ avec $\alpha = 3.23$	0.50
0.26	A_4^*	$\phi_{2,\alpha}$ avec $\alpha = 7.83$	0.70

Tableau 4.1: *Valeurs du paramètre α des générateurs $\phi_{1,\alpha}$ et $\phi_{2,\alpha}$, et valeurs du tau de Kendall pour les copules Archimax correspondantes.*

Nous avons donc deux plans d'expérience à trois facteurs, l'un correspondant à $\tau_{A^*} = 0.10$ et l'autre à $\tau_{A^*} = 0.26$. Dans ces deux plans, les facteurs sont

1) Les 7 estimateurs retenus, les trois du type Khoudraji, les trois du type Joe et al. et celui de Capéraà et al.;

2) Deux fonctions de dépendance, l'une symétrique et l'autre asymétrique;

3) Les 2 générateurs avec chacun deux valeurs du paramètre α.

Ces trois facteurs sont considérés comme des facteurs fixes, et pour chaque combinaison des niveaux de ces facteurs, 100 échantillons de taille 2500 ont été générés indépendamment. Pour l'analyse des données, nous procéderons à deux analyses de variance: la première utilisant comme variable dépendante la valeur du logarithme de la différence en valeur absolue entre l'estimation du tau de Kendall et la valeur du tau théorique, tandis que la deuxième utilisera la valeur du logarithme de L_1. L'utilisation de $log|\hat{\tau}_{A^\bullet} - \tau_{A^\bullet}|$ et de $log(L_1)$ plutôt que de $|\hat{\tau}_{A^\bullet} - \tau_{A^\bullet}|$ et de L_1 nous assure que l'on se trouve approximativement sous les hypothèses usuelles de l'analyse de variance. Remarquons toutefois que d'après les diagrammes en boîte de $log(L_1)$ apparaissant à l'annexe A, l'estimateur K_1 n'a pas la même variabilité que les autres. Nous le conservons quand même dans l'analyse. Nous ne présentons pas les diagrammes en boîte de $log|\hat{\tau}_{A^\bullet} - \tau_{A^\bullet}|$ qui sont semblables à ceux de $log(L_1)$.

Les analyses de variance effectuées sur les variables $log|\hat{\tau}_{A^\bullet} - \tau_{A^\bullet}|$ et $log(L_1)$ montrent que les interactions de deuxième et troisième ordre, faisant intervenir le deuxième facteur, ne sont pas significatives au niveau 0.05 et que ce facteur n'a pas d'influence. Il existe par contre une interaction significative entre les estimateurs et les générateurs, $\phi_{i,\alpha}$, $i = 1, 2$, c'est-à-dire entre le premier et le troisième facteur (niveau de signification observé: 0.0001). Les résultats obtenus pour les comparaisons multiples effectuées au niveau 0.05 entre les 7 estimateurs, lorsque la variable dépendante est $log|\hat{\tau}_{A^\bullet} - \tau_{A^\bullet}|$, sont résumés dans le Tableau 4.2, et ce pour chaque niveau du 3e facteur. Le symbole d'inégalité stricte signifie que les estimateurs sont significativement différents pour la variable réponse étudiée, tandis que le symbole $\approx$ signifie que les estimateurs ne le sont pas.

	ϕ *de Clayton*	
$\tau_{A^\bullet}$	α=4/7	α=1.6
0.10	$J_1 < K_3 < J_2 \approx J_3 \approx K_2 < CFG < K_1$	$J_1 < K_3 < J_2 < J_3 < K_2 < CFG < K_1$
$\tau_{A^\bullet}$	α=0.972	α=2.953
0.26	$J_1 \approx K_2 < K_3 < J_2 \approx J_3 < CFG \approx K_1$	$J_1 < K_2 \approx K_3 < J_2 \approx J_3 < CFG < K_1$
	ϕ *de Franck*	
$\tau_{A^\bullet}$	α=2.09	α=4.8
0.10	$J_1 < J_3 \approx K_3 < J_2 \approx K_2 < CFG < K_1$	$J_1 \approx J_3 \approx K_3 < J_2 < CFG \approx K_2 < K_1$
$\tau_{A^\bullet}$	α=3.23	α=7.83
0.26	$J_1 \approx K_2 \approx K_3 < J_2 < J_3 < CFG < K_1$	$J_1 \approx J_2 \approx K_2 \approx K_3 < J_3 < CFG < K_1$

Tableau 4.2: *Comparaisons multiples des sept estimateurs de la fonction de dépendance A pour différentes copules Archimax, et pour* $log|\hat{\tau}_{A^\bullet} - \tau_{A^\bullet}|$.

L'examen du tableau précédent permet immédiatement de constater que J_1, l'estimateur de Joe et al. avec le plus grand nombre de $T_i(\beta)$ retenus, est le meilleur dans tous les cas examinés. Les autres estimateurs de Joe, J_2 et J_3, semblent être comparables lorsque le générateur est de Clayton, à l'exception du cas où $\alpha = 1.6$, J_2 est alors plus précis que J_3. Lorsque le générateur utilisé est celui de Franck, on remarque que pour une force de dépendance petite, $\tau_{A^\bullet} = 0.10$, J_3 semble être meilleur que J_2. Par contre, lorsque la force de dépendance devient plus grande, $\tau_{A^\bullet} = 0.26$, J_2 domine J_3.

Pour ce qui est des trois estimateurs de Khoudraji, K_1, K_2 et K_3, on peut souligner immédiatement que celui possédant le plus petit seuil, $r = 0.01$, n'est pas un bon estimateur. Les deux autres, K_2 et K_3, semble se comporter différemment lorsque la force de dépendance augmente. En effet, lorsque $\tau_{A^\bullet} = 0.10$, l'estimateur K_3 est toujours meilleur que l'estimateur K_2. Par contre, lorsque $\tau_{A^\bullet}$ devient plus grand, $\tau_{A^\bullet} = 0.26$, les 2 estimateurs sont comparables pour la majorité des générateurs, à l'exception du cas où on utilise un générateur de Clayton avec $\alpha = 0.972$, K_2 étant alors meilleur que K_3. Si on compare maintenant ces deux derniers avec les estimateurs de Joe et al., on s'aperçoit qu'au moins un des estimateurs de Khoudraji, K_2 ou K_3, est le plus performant après l'estimateur J_1.

L'estimateur de Capéraà et al., CFG, ne donne pas de bons résultats comparativement aux autres à l'exception de K_1.

Dans le tableau qui suit, on trouve les comparaisons multiples entre les 7 estimateurs pour chaque niveau du troisième facteur, lorsque la variable dépendante est $log(L_1)$.

	ϕ *de Clayton*	
τ_{A^*}	α=4/7	α=1.6
0.10	$J_1 < J_2 \approx K_2 \approx K_3 < J_3 < CFG \approx K_1$	$J_1 \approx J_2 \approx K_3 < J_3 \approx K_2 < CFG < K_1$
τ_{A^*}	α=0.972	α=2.953
0.26	$J_1 < K_2 < J_2 \approx K_1 \approx K_3 < CFG \approx J_3$	$J_1 \approx K_2 < J_2 \approx K_3 < CFG \approx J_3 < K_1$
	ϕ *de Franck*	
τ_{A^*}	α=2.09	α=4.8
0.10	$J_1 < J_2 \approx K_2 \approx K_3 < J_3 < CFG < K_1$	$J_1 \approx J_2 \approx K_3 < J_3 < K_2 < CFG < K_1$
τ_{A^*}	α=3.23	α=7.83
0.26	$J_1 < J_2 \approx K_2 < K_3 < CFG \approx J_3 \approx K_1$	$J_1 \approx J_2 \approx K_2 \approx K_3 < J_3 < CFG < K_1$

Tableau 4.3: *Comparaisons multiples des sept estimateurs de la fonction de dépendance A pour différentes copules Archimax, à partir de* $log(L_1)$.

En examinant le Tableau 4.3, il apparait que J_1 est encore le meilleur estimateur. De plus, J_2 est, pour tous les attracteurs, meilleur que J_3, celui-ci étant un estimateur pauvre dans la majorité des cas, en particulier lorsque $\tau_{A^*} = 0.26$. Ces résultats sont également soutenus par les diagrammes en boîte fournis à l'Annexe A. Encore ici, les estimateurs de Khoudraji, K_2 et K_3, semblent avoir une certaine relation avec la force de dépendance. Lorsque $\tau_{A^*} = 0.10$, on remarque que K_3 domine K_2 dans la majorité des cas, tandis que pour $\tau_{A^*} = 0.26$, K_2 est meilleur que K_3. On note encore que les estimateurs CFG et K_1 ne sont pas de bons estimateurs.

Les graphiques représentant les estimations de A pour J_1, J_2, J_3, K_2 et K_3 ainsi que la fonction A théorique sont donnés à l'Annexe B. Ces graphiques confirment bel et bien les conclusions précédentes sur ces estimateurs.

Suite à ces différentes analyses, on peut conclure que, lorsque l'échantillon d'observations n'est pas constitué de valeurs extrêmes, l'estimateur J_1 de Joe et al. utilisant les 8% plus grandes pseudo-observations est nettement le meilleur estimateur parmi ceux étudiés ici. L'estimateur de Khoudraji, utilisé avec un seuil assez grand, peut aussi nous donner de bons résultats. Contrairement à la méthode d'estimation de Joe et al., le choix du seuil de l'estimateur de Khoudraji semble être un peu plus compliqué. En effet, les différentes analyses montrent que, pour obtenir un bon estimateur, le choix du seuil devrait tenir compte de la force de dépendance de l'attracteur. De plus, pour un seuil petit (0.01), l'estimation de A semble très mauvaise. Notons également que la condition exigeant l'indépendance entre les W_i et les R_i, $R_i > nr$, n'est pas suffisante pour nous éclairer dans ce choix puisque, pour chaque valeur du seuil utilisée, cette condition avait été préalablement vérifiée.

4.3.2 Analyse 2

Pour la deuxième analyse, nous avons simulé des observations provenant de copules de valeurs extrêmes. Nous avons effectué cette analyse pour de nouveau comparer les 3 méthodes d'estimation, mais surtout pour vérifier si l'estimateur de Capéraà et al. donnait de meilleurs résultats que les deux autres, étant donné que l'échantillon provient de valeurs extrêmes directement.

Pour comparer les estimateurs, nous avons considéré deux copules de valeurs extrêmes C_{A^*}, soit celle de Gumbel, symétrique par rapport à 1/2, dont la fonction de dépendance est donnée par

$$A_r^*(t) = \{t^r + (1-t)^r\}^{1/r},\ r \geq 1,$$

et celle asymétrique de Tawn (1988), dont la fonction de dépendance est donnée par

$$A_{\alpha,\beta,r}^*(t) = 1 - \beta + (\beta - \alpha)t + \{\alpha^r t^r + \beta^r (1-t)^r\}^{1/r},\ 1 \leq \alpha, \beta \leq 1,\ r \geq 1.$$

Plus précisément, les deux copules de valeurs extrêmes qui ont été simulées sont sous la forme

$$C_{A^*}(x,y) = exp[\{log(x) + log(y)\}A^*\{\frac{log(x)}{log(x)+log(y)}\}]$$

où les fonctions de dépendance sont données par A^*_r ou encore $A^*_{\alpha,\beta,r}$.

Ensuite, nous avons choisi trois forces de dépendance pour ces A^*, soit $\tau_{A^*} = 1/4, 1/2$ et $3/4$, ainsi que deux tailles d'échantillon, 50 et 100. Afin d'obtenir les trois forces de dépendance voulues, nous avons déterminé les valeurs des paramètres r, α et β que l'on peut trouver dans le tableau suivant.

τ_{A^*}	$A^*_r(t)$	$A^*_{\alpha,\beta,r}$	
1/4	r=4/3	r=1.42	$(\alpha,\beta) = (0.78, 0.97)$
1/2	r=2	r=2.58	$(\alpha,\beta) = (0.78, 0.97)$
3/4	r=4	r=50	$(\alpha,\beta) = (0.78, 0.97)$

Tableau 4.4: *Valeurs des paramètres de la copule de Gumbel, A^*_r, et de la copule asymétrique de Tawn, $A^*_{\alpha,\beta,r}$, pour obtenir les différentes forces de dépendance.*

Rappelons que dans cette analyse, nous comparons pour les trois forces de dépendance 5 estimateurs soit, J^*_1, J^*_2, K^*_1, K^*_2 et CFG (voir Section 4.2). Cette comparaison se fera selon un plan à quatre facteurs qui sont

1) Les 5 estimateurs, deux du type Khoudraji, deux du type Joe et al. et celui de Capéraà et al.;

2) Les 2 fonctions de dépendance, A^*_r et $A^*_{r,\alpha,\beta}$;

3) Les 3 forces de dépendance, $\tau_{A^*} = 1/4$, $\tau_{A^*} = 1/2$ et $\tau_{A^*} = 3/4$;

4) Les 2 tailles d'échantillon, 50 et 100.

Ces quatre facteurs sont considérés comme des facteurs fixes, et pour chaque combinaison de niveaux de ces facteurs, 100 échantillons ont été générés indépendamment. Comme pour l'analyse 1, nous procéderons à deux analyses de variance: la première utilisant la valeur du logarithme de la différence en valeur absolue entre l'estimation du tau de Kendall et la valeur du tau théorique comme variable dépendante, tandis que la deuxième utilisera la valeur du logarithme de L_1. L'utilisation de $log|\hat{\tau}_{A^*} - \tau_{A^*}|$ et de $log(L_1)$ plutôt que de $|\hat{\tau}_{A^*} - \tau_{A^*}|$ et de L_1 nous assure que l'on se trouve approximativement sous les hypothèses usuelles de l'analyse de variance.

L'analyse de variance sur la variable $log|\hat{\tau}_{A^*} - \tau_{A^*}|$ montre qu'il existe une interaction significative de troisième ordre (niveau de signification observé: 0.0101) entre le premier, le troisième et le quatrième facteur. De plus, l'interaction entre le deuxième et le troisième facteur est également significative (niveau de signification observé: 0.0001). L'analyse de variance sur la variable $log(L_1)$ permet de voir qu'il existe une interaction significative de quatrième ordre. Les comparaisons des estimateurs seront effectuées sur chaque combinaison des niveaux des trois derniers facteurs.

Dans le Tableau 4.5, on trouve les résultats obtenus pour les comparaisons multiples au niveau 0.05 entre les 5 estimateurs, et ce pour chaque combinaison des niveaux des facteurs 2, 3 et 4, pour la variable dépendante $log|\hat{\tau}_{A^*} - \tau_{A^*}|$.

	symétrie	
τ_{A^*}	N=50	N=100
0.25	$CFG < J_1^* \approx J_2^* \approx K_2^* < K_1^*$	$CFG < J_1^* \approx J_2^* \approx K_1^* \approx K_2^*$
0.50	$CFG \approx K_1^* < J_1^* \approx J_2^* \approx K_2^*$	$CFG < K_1^* < J_1^* \approx J_2^* \approx K_2^*$
0.75	$CFG < K_1^* < J_1^* < J_2^* \approx K_2^*$	$CFG < J_1^* \approx K_1^* < J_2^* \approx K_2^*$
	asymétrie	
τ_{A^*}	N=50	N=100
0.25	$CFG < J_1^* \approx J_2^* \approx K_1^* \approx K_2^*$	$CFG < J_1^* \approx J_2^* \approx K_1^* \approx K_2^*$
0.50	$CFG \approx K_1^* < J_1^* \approx K_2^* < J_2^*$	$CFG < J_1^* \approx K_1^* \approx K_2^* < J_2^*$
0.75	$CFG \approx K_1^* < J_1^* \approx J_2^* \approx K_2^*$	$CFG < J_1^* \approx J_2^* \approx K_1^* \approx K_2^*$

Tableau 4.5: *Comparaisons multiples des cinq estimateurs de la fonction de dépendance A^* de différentes copules de valeurs extrêmes, pour différentes tailles d'échantillon N et pour $log|\hat{\tau}_{A^*} - \tau_{A^*}|$.*

L'étude du tableau précédent met clairement en évidence le fait que dans les cas considérés, l'estimateur CFG est le meilleur.

Pour ce qui est des estimateurs de Khoudraji, K_1^* et K_2^*, on remarque que, lorsque la force de dépendance est de 0.50 ou de 0.75, K_1^* est meilleur que K_2^*, à l'exception du cas où $N = 100$ avec une fonction de dépendance asymétrique. Pour ce cas, les estimateurs semblent être comparables. Par contre, lorsque la force de dépendance diminue, c'est-à-dire lorsque $\tau_{A^*} = 0.25$, on ne remarque pas de différence entre les 2 estimateurs, sauf peut-être lorsque $N = 50$ avec A^* symétrique, K_2^* étant alors meilleur que K_1^*. Encore ici, la performance des estimateurs de Khoudraji semble être reliée avec la force de dépendance. Soulignons également que le meilleur estimateur après celui de CFG est, pour tous les cas, un estimateur de Khoudraji.

Les estimateurs J_1^* et J_2^* donnent des résultats semblables pour tous les cas. Mentionnons par contre que, pour le cas où A^* est symétrique et $\tau_{A^*} = 0.75$, J_1^* est meilleur que J_2^*, de même que pour le cas où A^* est asymétrique et $\tau_{A^*} = 0.50$.

Les résultats obtenus pour les comparaisons multiples portant sur la variable $log(L_1)$ sur chaque combinaison des niveaux des facteurs 2, 3 et 4 sont présentés dans le tableau suivant.

	symétrie	
τ_{A^*}	N=50	N=100
0.25	$CFG < J_2^* \approx K_1^* \approx K_2^* < J_1^*$	$CFG < J_1^* \approx K_1^* \approx K_2^* < J_2^*$
0.50	$CFG < K_1^* < K_2^* < J_2^* < J_1^*$	$CFG < K_1^* < J_1^* \approx K_2^* < J_2^*$
0.75	$CFG < K_1^* < K_2^* < J_2^* < J_1^*$	$CFG < K_1^* < K_2^* < J_1^* < J_2^*$
	asymétrie	
τ_{A^*}	N=50	N=100
0.25	$CFG < K_1^* \approx K_2^* < J_2^* < J_1^*$	$CFG < K_1^* < J_1^* \approx K_2^* < J_2^*$
0.50	$CFG < K_1^* < K_2^* < J_2^* < J_1^*$	$CFG < K_1^* < J_1^* \approx J_2^* \approx K_2^*$
0.75	$CFG < K_1^* < K_2^* < J_2^* < J_1^*$	$CFG < K_1^* < J_1^* \approx J_2^* \approx K_2^*$

Tableau 4.6: *Comparaisons multiples de cinq estimateurs de la fonction de dépendance A^* de différentes copules de valeurs extrêmes, pour différentes tailles d'échantillon N et pour $log(L_1)$.*

Encore ici, il est évident que l'estimateur CFG est de loin le plus précis parmi les estimateurs étudiés. Ce résultat est également vérifié à partir des diagrammes en boîte des erreurs $log(L_1)$ fournis à l'Annexe C ainsi que des graphiques représentant les estimations de la fonction de dépendance et la valeur théorique de A^* fournis à l'Annexe D.

Si on examine les estimateurs K_1^* et K_2^*, on s'aperçoit encore que K_1^* est meilleur que K_2^* lorsque la force de dépendance est de 0.50 ou de 0.75, mais qu'ils deviennent comparables lorsque la force de dépendance est de 0.25. De plus, ces deux estimateurs dominent ceux de Joe et al.

Pour ce qui est de J_1^* et de J_2^*, on s'aperçoit que lorsque la taille d'échantillon est de 50, J_2^* est toujours meilleur que J_1^*, tandis que pour une taille d'échantillon de 100, J_1^* domine J_2^*.

On peut conclure par cette analyse que l'estimateur de Capéraà et al. est le meilleur estimateur lorsque les observations sont des valeurs extrêmes. De plus, l'estimateur de Khoudraji calculé avec un nombre suffisant d'observations peut aussi être un bon estimateur de la fonction de dépendance. Par contre, la méthode de Joe et al. est à proscrire pour ce type d'observations.

De ces deux analyses, portant sur des valeurs extrêmes et non extrêmes, nous retenons tout d'abord que l'estimateur de Capéraà et al. est l'estimateur à utiliser lorsque l'échantillon est constitué de valeurs extrêmes. Celui de Khoudraji peut aussi nous fournir de bonnes estimations pour ce cas, mais il est nécessaire de s'assurer que le nombre d'observations utilisées est suffisant.

Pour le cas où les observations ne sont pas extrêmes, l'estimateur de Joe et al. calculé avec les 8% plus grandes pseudo-observations T_i est sans aucun doute le meilleur. Celui de Khoudraji peut aussi nous permettre d'obtenir de bons résultats, mais le choix du seuil est difficile à déterminer puisqu'il semble que ce choix devrait être fait selon la force de dépendance de l'attracteur C_{A^*}.

Chapitre 5

Analyse de données

Dans le chapitre précédent, nous avons comparé différentes méthodes d'estimation de la fonction de dépendance, soit celles de Capéraà et al., de Joe et al., ainsi que de Khoudraji. Ces comparaisons ont été effectuées, dans un premier temps à partir de simulations d'échantillons provenant de lois bidimensionnelles ayant un attracteur donné, et dans un deuxième temps à partir de simulations d'échantillons de lois bivariées de valeurs extrêmes. Des deux analyses effectuées, nous avons conclu que l'estimateur de Capéraà et al. était nettement supérieur aux deux autres méthodes d'estimation, et ce lorsque l'échantillon de départ était constitué de valeurs extrêmes. Par contre, cet estimateur ne donnait pas de bons résultats lorsque l'échantillon utilisé était constitué de couples de lois bidimensionnelles. Pour ce cas, la première analyse du chapitre précédent a montré que l'estimateur de Joe construit avec les 200 plus grandes pseudo-observations $T_i(\beta)$ était le meilleur. De plus, lorsqu'on compare les trois estimateurs de Khoudraji, ceux utilisant un seuil de 0.08 et de 0.04 donnent de meilleurs résultats que celui construit avec un seuil de 0.01.

Dans ce chapitre, nous appliquons ces estimateurs à un ensemble de données concernant des réservoirs d'eau utilisés pour produire l'électricité nécessaire à l'entreprise Alcan. Nous allons expliquer brièvement la façon dont sont obtenues ces données telle qu'elle est présentée dans le rapport "Validation des apports non contrôlés historiques" rédigé par l'Institut national de la recherche scientifique, INRS-Eau.

Les données représentent la quantité d'eau qu'un réservoir reçoit à l'état naturel durant un intervalle de temps déterminé. Cette quantité est appelée l'apport naturel. Cette valeur tient compte de différents paramètres comme les précipitations sur le réservoir, le ruissellement du bassin de drainage qui l'entoure, l'évaporation à sa surface et l'écoulement souterrain. Puisque ces différents paramètres ne peuvent être mesurés directement, Hydro-Québec utilise une équation qui permet de calculer l'apport naturel, A_n, appelée équation du bilan hydrique:

$$A_n = Q_{Av} - Q_{Am} + \delta S$$

où Q_{Av} représente les débits d'eau à la sortie du réservoir pendant 24 heures, Q_{Am}, les débits d'eau provenant d'un réservoir en amont pendant 24 heures et δS, la variation du volume stocké dans le réservoir pendant 24 heures.

Les apports naturels sont calculés sur près de 45 réservoirs majeurs, à des sites prédéterminés. Naturellement, ces mesures sont quelque peu entachées d'erreurs, qui peuvent être contrôlées par un procédé de filtrage des données, utilisé par Hydro-Québec. Pour notre étude, nous avons choisi d'utiliser les données des apports naturels sur les réservoirs Chute du Diable et Lac St-Jean car, selon Hydro-Québec, ces mesures contiennent moins d'erreurs que celles sur les autres bassins. Notre objectif est d'estimer non paramétriquement la fonction de dépendance de la loi bivariée des maxima des apports naturels de ces deux réservoirs. Cette estimation peut servir à choisir un modèle paramétrique pour cette loi. Les réservoirs sont représentés sur les cartes à l'Annexe E.

Les apports naturels sur ces deux bassins nous ont été fournis suivant deux ensembles. Le premier contient les maxima annuels des apports naturels pour une période allant de 1953 à 1994, tandis que le deuxième contient les valeurs journalières des apports naturels pour les années 1953 à 1993. Puisque les maxima des apports naturels ont été pris sur la période du printemps, soit du 1er avril au 30 juin, nous utiliserons seulement les valeurs journalières sur cette période.

A partir de ces données, nous utiliserons dans un premier temps, l'estimateur de Capéraà et al., qui sera calculé à partir de l'ensemble de données formé des maxima annuels des apports naturels. Nous obtiendrons ainsi l'estimation de la fonction de dépendance de la loi de ces maxima et également le tau de Kendall calculé à partir de cette estimation. Dans un deuxième temps, les estimateurs de Joe et al. et de Khoudraji seront appliqués à l'ensemble des données journalières des apports naturels. Nous obtiendrons ainsi les estimations de la fonction de dépendance de l'attracteur associé à ces données. Cependant, comme il est difficile de comparer graphiquement ces estimations, nous ferons ces comparaisons au moyen des tau de Kendall calculés à partir de ces estimations.

5.1 Estimation par la méthode de Capéraà, Fougères et Genest sur les apports naturels maximums des bassins de l'Alcan.

L'ensemble de données dont nous disposons est constitué des maxima annuels des apports naturels sur les années 1953 à 1994, ce qui forme donc un échantillon de 42 couples (X_i, Y_i) de valeurs extrêmes, où X_i représente le maximum sur les apports naturels journaliers pour l'année i sur le bassin "Lac St-Jean", et Y_i, le maximum pour la même année sur le bassin "Chute du Diable".

Nous allons estimer la fonction de dépendance de la loi de ces maxima bidimensionnels par la méthode proposée par Capéraà et al. Comme cette méthode est applicable essentiellement à des lois de valeurs extrêmes, nous allons tout d'abord vérifier que cet échantillon provient d'une des lois de ce type. Pour ce faire, nous utiliserons un test construit par Ghoudi, Khoudraji et Rivest (1997) décrit ci-dessous.

Soit $\{(X_i, Y_i) : i = 1, ..., n\}$ un échantillon bivarié d'une loi H. Pour vérifier si H appartient à la famille des lois de valeurs extrêmes bidimensionnelles, il est nécessaire de s'assurer que la copule C, associée à H, appartient à la famille des copules de valeurs extrêmes, c'est-à-dire a la forme suivante

$$C(u, v) = exp[\{log(u) + log(v)\}A\{\frac{log(u)}{log(u) + log(v)}\}], 0 < u, v < 1, \qquad (5.1)$$

où A est la fonction de dépendance que l'on veut estimer. Posons $W = C(U, V)$. Si $C(x, y)$ a la forme (5.1), alors $8E(W) - 9E(W^2) - 1 = 0$. Soit S_n une estimation non-biaisée de $8E(W) - 9E(W^2) - 1$ (voir Ghoudi et al., 1997, pour l'expression de S_n). L'hypothèse H_0 : $C(x, y)$ a la forme (5.1), sera rejetée lorsque $|S_n|/\{\hat{V}_{jack}(S_n)\}^{1/2} > z_{1-\alpha/2}$, où $\hat{V}_{jack}(S_n)$ représente une estimation de la variance de S_n obtenue par la méthode du jackknife et $z_{1-\alpha}$, le quantile d'ordre $1 - \alpha$ de la loi normale centrée réduite. Ainsi, on peut conclure que si l'hypothèse H_0 n'est par rejetée, la copule C appartient à la famille des copules de valeurs extrêmes. Pour obtenir les détails sur ce test, voir Ghoudi, Khoudraji et Rivest (1997).

En effectuant ce test sur l'ensemble de données composé des maxima annuels bidimensionnels des apports naturels journaliers, nous avons obtenu la valeur 1.95 pour $|S_n|/\{\hat{V}_{jack}(S_n)\}^{1/2}$, ce qui est légèrement inférieur au quantile d'ordre $1 - \alpha/2$ de la loi normale centrée réduite lorsque α est de 0.05. Ainsi n'ayant pas rejeté l'hypothèse H_0, nous pouvons conclure que la copule C associée à la loi bivariée de l'échantillon est une copule de valeurs extrêmes.

Puisque les lois marginales de cet échantillon ne sont pas connues, nous avons utilisé l'estimateur $A_n^*(t)$ donné à la Section 3.1. Le graphique suivant montre l'estimation de la fonction de dépendance A.

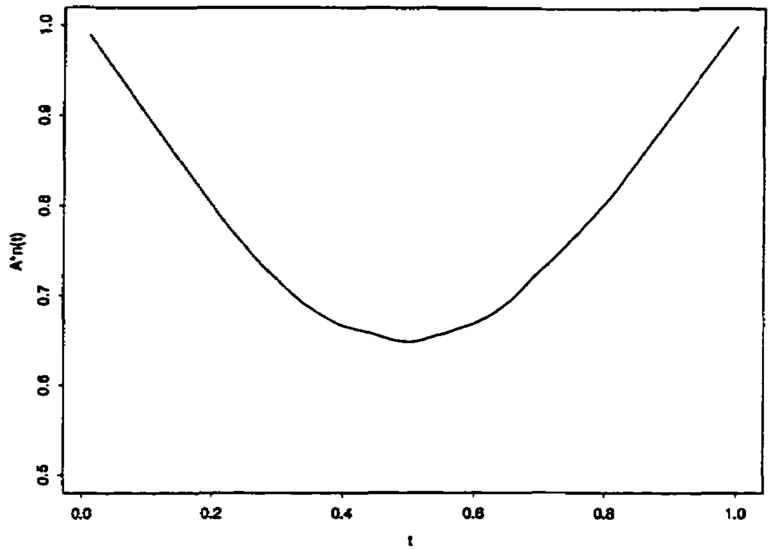


Figure 5.1: *Estimation de la fonction de dépendance A calculée sur l'ensemble des maxima par la méthode de Capéraà et al.*

Nous savons que $A_n^*(t)$ est un bon estimateur de la fonction de dépendance A d'une loi de valeurs extrêmes. Or le test effectué plus haut pour vérifier si l'échantillon des maxima provenait d'une copule extrême avait un seuil de signification observé très proche du seuil théorique de 0.05 que l'on s'était fixé. Aussi, ne savons-nous pas très bien si l'on a estimé la fonction de dépendance d'une loi des valeurs extrêmes. Cependant, on peut en avoir une idée en comparant les estimations du tau de Kendall obtenues, d'une part à partir de $A_n^*(t)$, et d'autre part à partir de l'échantillon des maxima. Denotons ces estimations par $\tau_{A_n^*}$ et $\hat{\tau}_n$ respectivement. On trouve $\tau_{A_n^*} = 0.6310$ et $\hat{\tau}_n = 0.7097$. Comment comparer ces deux valeurs? Connnaissant la loi asymptotique de $\hat{\tau}_n$ (Genest et Rivest, 1993), il est possible de déterminer un intervalle de confiance pour la valeur théorique τ_A du tau de Kendall.

Cet intervalle, de niveau $1-\alpha$, est construit de la façon suivante.

Soit (X_i, Y_i), $i = 1, .., n$, les couples de valeurs extrêmes. Posons

$$V_i = card\{(X_j, Y_j) < (X_i, Y_i)\}/(n-1)$$

et

$$W_i = card\{(X_j, Y_j) > (X_i, Y_i)\}/(n-1).$$

L'intervalle de confiance est le suivant:

$$\hat{\tau_n} \pm z_{1-\alpha/2}\frac{\hat{\gamma}_n}{n^{1/2}}$$

où $\hat{\gamma}_n$ est l'estimateur de la variance de $\hat{\tau}_n$ donné par (voir Genest et Rivest, 1993) $\hat{\gamma_n^2} = 16\sum_{i=1}^{n}\frac{(V_i+W_i-2\bar{V})^2}{n-1}$, avec $\bar{V} = \sum_{i=1}^{n}\frac{V_i}{n}$, et où $z_{1-\alpha}$ est le quantile d'ordre $1-\alpha$ de la loi normale centrée réduite.

L'intervalle de niveau $1-\alpha = 0.95$ est alors [0.6155,0.8040]. Puisque $\tau_{A_n^*}$ est inclus dans cet intervalle, ceci suggère que l'estimation A_n^* est celle de la fonction de dépendance d'une loi des valeurs extrêmes. Cette estimation peut alors servir à ajuster un modèle paramétrique pour les maxima annuels.

5.2 Estimation par les méthodes de Joe et al. et de Khoudraji sur l'ensemble des apports naturels journaliers des bassins de l'Alcan

Dans cette section, nous allons calculer les estimateurs de Joe et al. et de Khoudraji sur les données journalières afin d'estimer la fonction de dépendance A des maxima annuels. Pour chaque année j, nous avons 91 données journalières (X_{ij}, Y_{ij}) s'étalant du 1er avril au 30 juin, où X_{ij} représente l'apport naturel journalier sur le bassin Lac St-Jean, et Y_{ij}, l'apport naturel journalier sur le bassin Chute du Diable, pour le jour i à l'année j. Les méthodes de Joe et al. et de Khoudraji nécessitant un assez grand nombre de données puisqu'elles n'utilisent que les données supérieures à certains seuils, nous avons regroupé les années en quatre ensembles. Ainsi, les trois premiers ensembles seront formés des couples (X_{ij}, Y_{ij}) représentant les données journalières du printemps pour les années 1953 à 1962, 1963 à 1972 et 1973 à 1982. Ces échantillons seront appelés respectivement échantillon 1,2 et 3. Le nombre total d'observations par échantillon sera donc de 910 couples de données, c'est-à-dire 91 couples (X_{ij}, Y_{ij}) par année, et ce sur 10 ans. Le dernier échantillon, que l'on appellera échantillon 4, sera constitué des apports naturels journaliers pour les années 1983 à 1993, soit 91 couples (X_{ij}, Y_{ij}) sur 11 ans, pour un total de 1001 couples de données. Les estimateurs de A seront donc calculés à partir de chacun des 4 échantillons de données pour chacune des méthodes d'estimation de Joe et al. et de Khoudraji.

5.2.1 Estimateur de Joe et al.

Nous avons vu au chapitre précédent que l'estimateur noté J_1 de Joe et al. utilisant les 8% plus grandes pseudo-observations $T_i(\beta)$, était le meilleur dans toutes les situations examinées. Aussi conservons-nous ici ce pourcentage. Le nombre $\tilde{n}$ de $T_i(\beta)$ retenus pour les 4 échantillons est donné par le Tableau 5.1. Rappelons de plus qu'une condition essentielle pour obtenir un bon estimateur consiste à vérifier si les valeurs $iT_{[i]}(\beta)$ qui interviennent dans le calcul de la moyenne tronquée sont constantes. Ainsi le nombre $\hat{n}$ d'observations que nous avons décidé de conserver pour calculer cette moyenne a été déterminé à partir des graphiques de la Figure (5.2) représentant les valeurs $iT_{[i]}(99)$, et ce pour les échantillons de couples 1, 2, 3 et 4 respectivement. Ce nombre $\hat{n}$ apparait dans le Tableau 5.1.

n	$\tilde{n}$	$\hat{n}$
910	73	63
1001	80	60

Tableau 5.1: *Valeurs $\tilde{n}$ du nombre représentant les 8% plus grandes pseudo-observations $T_i(\beta)$, ainsi que les valeurs $\hat{n}$ du nombre d'observations conservées pour le calcul de la moyenne tronquée, et ce pour chaque taille d'échantillon n.*

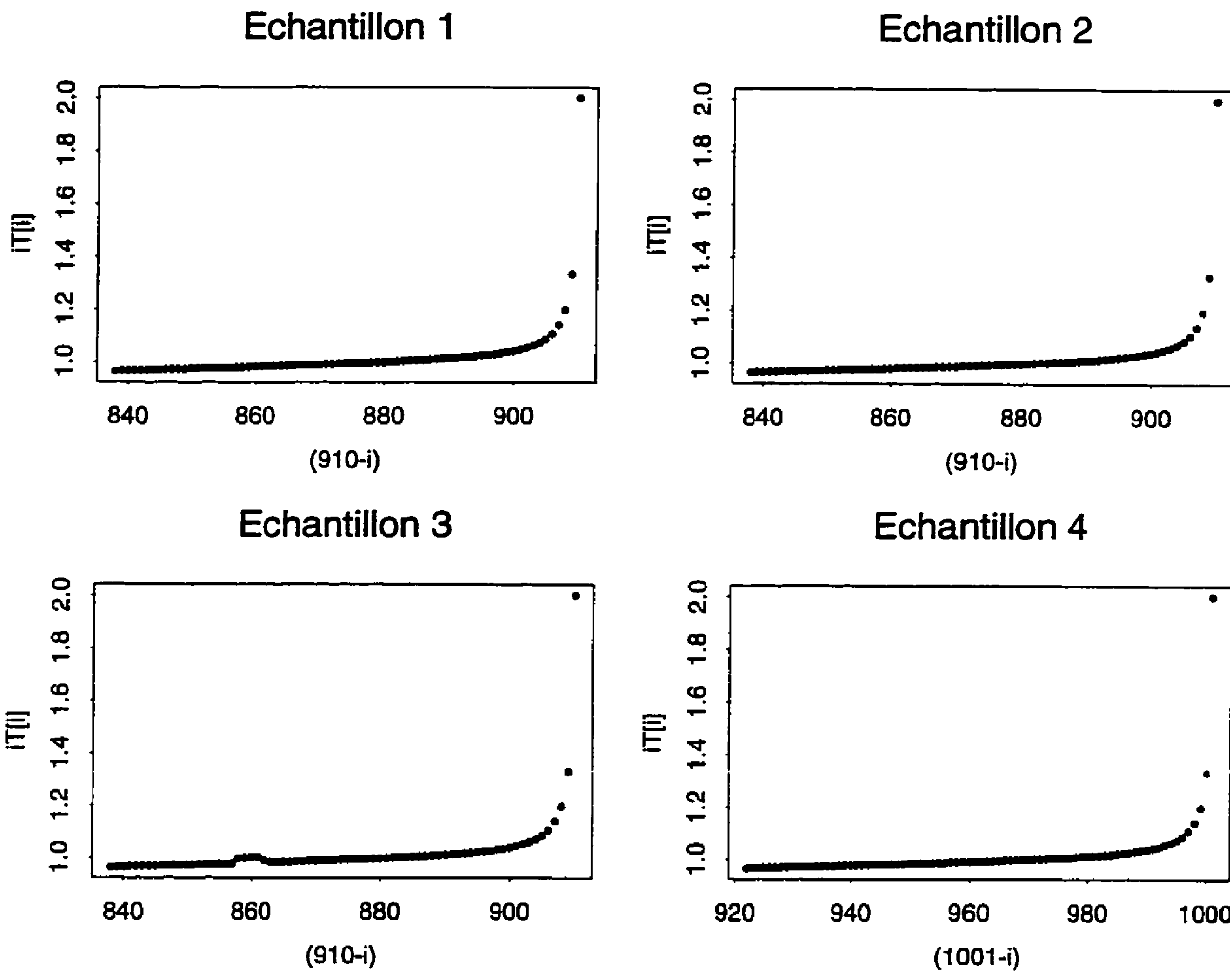


Figure 5.2: *Graphiques des valeurs* $iT_{[i]}(99)$ *calculées pour les échantillons 1, 2, 3 et 4.*

5.2.2 Estimateur de Khoudraji

Nous avons vu à la Section 3.2 que pour appliquer l'estimateur de Khoudraji, le seuil r devait être choisi de façon à ce que les W_i et les R_i, $R_i > nr$, soient indépendants. Pour chacun des 4 échantillons de couples bivariés et pour chaque seuil utilisé, nous avons vérifié que cette condition était respectée en calculant les valeurs du tau de Kendall et en effectuant un test sur l'hypothèse $H_0 : \tau = 0$ de niveau 0.05. Le Tableau 5.2 présente pour chacun des 4 échantillons, la valeur du tau de Kendall calculée entre les W_i et les R_i ainsi que le niveau de signification observé $\hat{\alpha}$ du test $H_0 : \tau = 0$, et ce pour chaque seuil que nous avons utilisé. Ce tableau présente également le nombre n^{**} de W_i supérieurs aux différents seuils.

numéro de l'échantillon	r	τ	$\hat{\alpha}$	n^{**}
1	25/910	-0.05021	0.5395	70
2	25/910	0.02777	0.7320	71
3	25/910	0.00108	0.9891	75
4	25/1001	-0.05485	0.4743	79
1	50/910	-0.07000	0.5376	38
2	50/910	-0.08254	0.4788	36
3	50/910	0.04453	0.6897	39
4	50/1001	0.00971	0.9250	43
1	100/910	-0.2223	0.2376	16
2	100/910	0.02924	0.8611	19
3	100/910	-0.06719	0.6534	23
4	100/1001	-0.07368	0.6497	20

Tableau 5.2: *Valeurs du tau de Kendall τ calculé entre les W_i et les R_i, du niveau de signification observé $\hat{\alpha}$ pour le test $H_0 : \tau = 0$ avec $\alpha = 0.05$, ainsi que du nombre n^{**} de W_i tel que $R_i > nr$, pour chacun des 4 échantillons.*

Puisque $\hat{\alpha}$ pour chaque seuil et pour chaque échantillon est nettement supérieur à 0.05, on peut dire que les W_i et les R_i sont indépendants.

5.2.3 Résultats sur l'ensemble des apports naturels journaliers

Les graphiques présentés aux Figures 5.3 à 5.6 montrent les estimations de la fonction de dépendance obtenues par l'estimateur de Joe et al. et les trois estimateurs de Khoudraji, et ce sur les quatres échantillons respectivement.

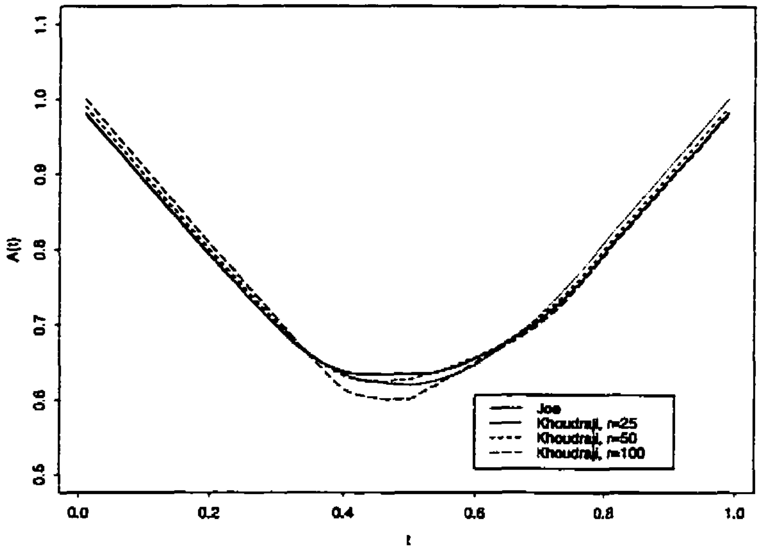


Figure 5.3: *Estimation de la fonction de dépendance, obtenue à partir de l'échantillon 1 d'apports naturels journaliers, par l'estimateur de Joe et al. et les estimateurs de Khoudraji.*

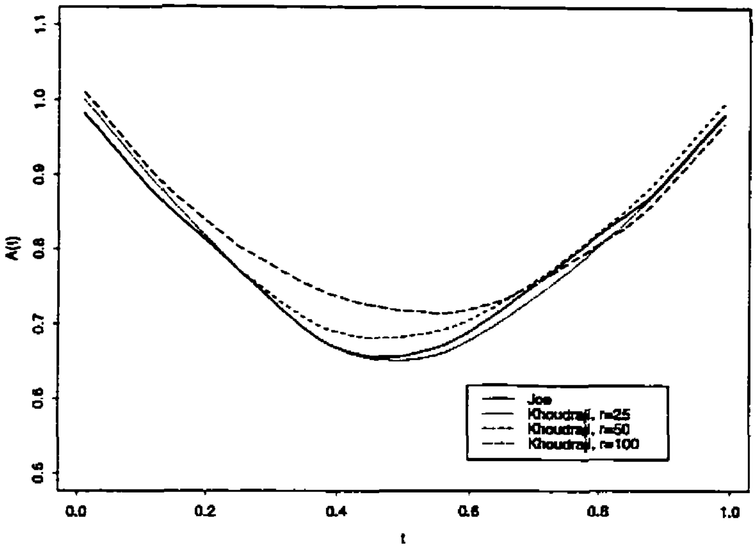


Figure 5.4: *Estimation de la fonction de dépendance, obtenue à partir de l'échantillon 2 d'apports naturels journaliers, par l'estimateur de Joe et al. et les estimateurs de Khoudraji.*

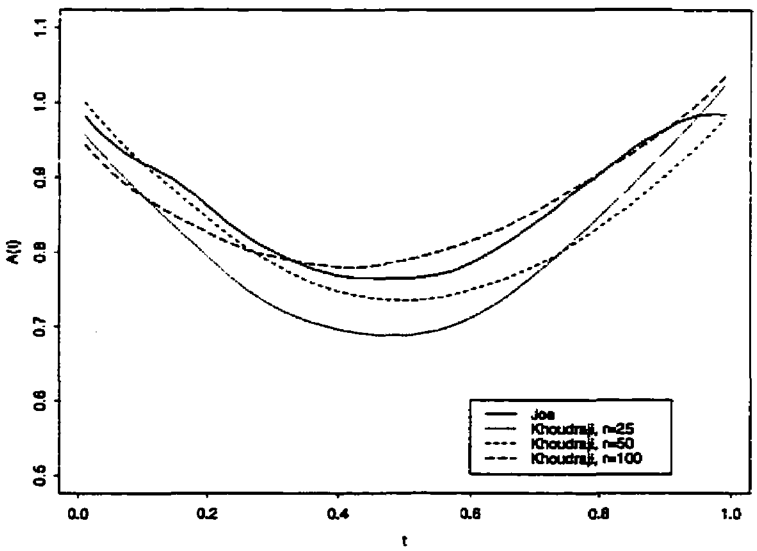


Figure 5.5: *Estimation de la fonction de dépendance, obtenue à partir de l'échantillon 3 d'apports naturels journaliers, par l'estimateur de Joe et al. et les estimateurs de Khoudraji.*

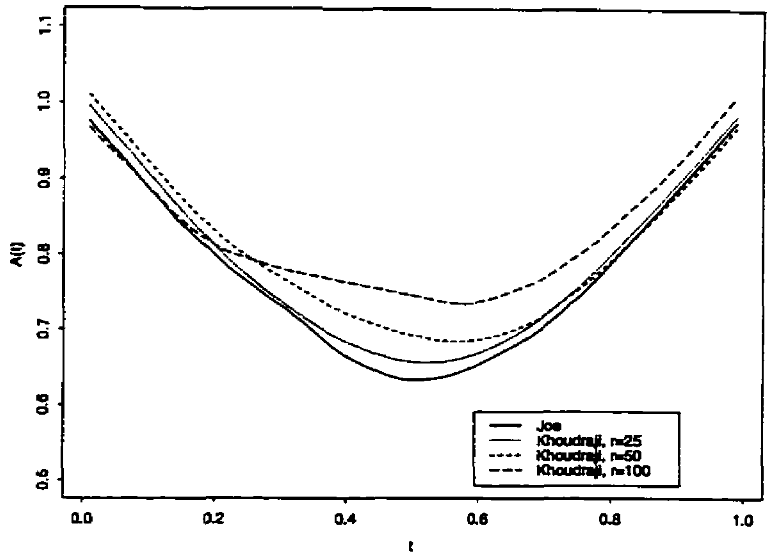


Figure 5.6: *Estimation de la fonction de dépendance, obtenue à partir de l'échantillon 4 d'apports naturels journaliers, par l'estimateur de Joe et al. et les estimateurs de Khoudraji.*

Par ces graphiques, on peut immédiatement remarquer que l'estimateur de Khoudraji fournissant l'estimation de la fonction de dépendance la plus comparable à celle obtenue par la méthode de Joe et al. n'est pas le même pour tous les échantillons. En effet, pour l'échantillon 1, l'estimateur de Khoudraji avec un seuil de 50/910 semble être celui qui fournit l'estimation la plus similaire à celle de Joe et al. Par contre, pour l'échantillon 2 et 4, celui utilisant un seuil de 25/910 semble être le meilleur et enfin, pour l'échantillon 3, l'estimateur utilisant un seuil de 100/910 est alors le plus comparable.

Les Tableaux 5.3 à 5.6 représentent respectivement les valeurs du tau de Kendall calculées sur les 4 échantillons pour les différents estimateurs de Joe et al. et de Khoudraji.

Methode de Joe et al.	*Methode de Khoudraji*	
$\tau_{\hat{A}}$	r	$\tau_{\hat{A}}$
0.674306187	25/910	0.7014584
	50/910	0.69319716
	100/910	0.74139605

Tableau 5.3: *Valeurs du tau de Kendall obtenues pour les différentes estimations de la fonction de dépendance effectuées à partir de l'échantillon 1 pour les méthodes de Joe et al. et de Khoudraji.*

Methode de Joe et al.	*Methode de Khoudraji*	
$\tau_{\hat{A}}$	r	$\tau_{\hat{A}}$
0.584366387	25/910	0.615610123
	50/910	0.552635004
	100/910	0.487820362

Tableau 5.4: *Valeurs du tau de Kendall obtenues pour les différentes estimations de la fonction de dépendance effectuées à partir de l'échantillon 2 pour les méthodes de Joe et al. et de Khoudraji.*

Methode de Joe et al.	*Methode de Khoudraji*	
$\tau_{\hat{A}}$	r	$\tau_{\hat{A}}$
0.336900572	25/910	0.538896002
	50/910	0.438216808
	100/910	0.3408514767

Tableau 5.5: *Valeurs du tau de Kendall obtenues pour les différentes estimations de la fonction de dépendance effectuées à partir de l'échantillon 3 pour les méthodes de Joe et al. et de Khoudraji.*

Methode de Joe et al.	*Methode de Khoudraji*	
$\tau_{\hat{A}}$	r	$\tau_{\hat{A}}$
0.632425003	25/1001	0.609368303
	50/1001	0.5458161942
	100/1001	0.449695978

Tableau 5.6: *Valeurs du tau de Kendall obtenues pour les différentes estimations de la fonction de dépendance effectuées à partir de l'échantillon 4 pour les méthodes de Joe et al. et de Khoudraji.*

En examinant les tableaux précédents, on remarque que les estimations des tau de Kendall obtenues par la méthode de Joe et al. sont très différentes pour les 4 échantillons. La loi des maxima semble donc avoir changé au cours du temps, ce qui ne correspond en rien aux maxima annuels.

Pour ce qui est des estimateurs de Khoudraji, on peut voir que les tau de Kendall les plus proches de ceux obtenus par l'estimateur de Joe et al. ne sont pas les mêmes pour chaque échantillon. Ici encore, on remarque que les estimations des tau de Kendall les plus comparables à celles de Joe et al. sont atteintes lorsque le seuil est de 50/910 pour l'échantillon 1, 25/910 pour l'échantillon 2 et 4, et enfin 100/910 pour l'échantillon 3, ce que nous avions déjà noté à partir des graphiques précédents.

Nous remarquons aussi que plus la force de dépendance estimée par la méthode de Joe et al. semble être forte, plus l'estimation de A devient semblable pour les trois estimateurs de Khoudraji. A l'inverse, lorsque la force de dépendance diminue, les estimations de A par les méthodes de Khoudraji sont alors très différentes entre elles et de celle obtenue par la méthode de Joe et al.

En résumé, ayant pris l'estimateur J_1 de Joe et al. comme référence, puisque d'après les résultats des simulations présentés au chapitre précédent il dominait les autres dans toutes les situations examinées, nous lui avons comparé ceux de Khoudraji. Il ressort de cette étude sur les données de l'Alcan que le choix du seuil pour appliquer la méthode de Khoudraji est très important mais difficile à faire. La raison en est que ce choix doit tenir compte de la force de dépendance de l'attracteur de la loi des observations, attracteur que l'on veut estimer. Aussi, tant que l'on n'aura pas trouvé une façon de choisir le seuil autre que celle s'appuyant sur l'indépendance entre les W_i et les R_i, nous suggérons d'estimer la fonction de dépendance A de l'attracteur par la méthode de Joe et al. utilisée dans la présente étude.

Chapitre 6

Conclusion

Dans ce travail, nous avons comparé trois méthodes d'estimation de la fonction de dépendance des copules de valeurs extrêmes bidimensionnelles, la première est une extension de celle de Capéraà et al., et les deux autres sont celles de Khoudraji et de Joe et al. Pour ceci nous avons envisagé les deux situations suivantes: soit nous disposons d'un échantillon d'une loi bidimensionnelle appartenant au domaine d'attraction d'une loi extrême, soit nous disposons d'observations de cette loi extrême. Dans la première situation, il ressort nettement de nos simulations que le meilleur estimateur est celui de Joe et al. utilisant les 8% plus grandes pseudo-observations $T_i(\beta)$. L'approche de Khoudraji donne de bons résultats mais on se heurte à la difficulté du choix du seuil qui doit être fait en fonction de la force de dépendance de la loi extrême que l'on ne connait pas. Son utilisation n'est donc pas recommandée dans ces conditions. L'estimateur de Capéraà et al. ne donne pas de bons résultats tel qu'appliqué dans cette première situation. A l'inverse, lorsque les données sont des maxima, l'estimateur de Capéraà et al. est le meilleur pour tous les cas que nous avons envisagés. Le comportement de l'estimateur de Khoudraji dépend encore de la force de dépendance de la loi extrême, mais dans cette situation il peut être appliqué car il est possible d'estimer cette dépendance directement à partir des observations. Quant à l'estimateur de Joe et al. il ne donne pas de bons résultats, cela étant dû probablement aux petits nombres de pseudo-observations retenues pour le calculer. Enfin, l'application de ces estimateurs aux données de l'Alcan a montré encore une fois que les résultats obtenus par la méthode de Khoudraji étaient très différents selon le choix du seuil. Aussi, est-il nécessaire d'étudier une autre façon de déterminer le seuil pour appliquer avec succès cet estimateur.

Bibliographie

Bortkiewicz, L. von (1922). Variationsbreite und mittlerer Fehler. *Sitzungsberichte Berliner Math. Ges.* **21.**

Capéraà, P., Fougères, A.-L. & Genest, C. (1997a). A Nonparametric Estimation Procedure for Bivariate Extreme Value Copulas. *Biometrika,* **84**, 3, 567–577.

Capéraà, P., Fougères, A.-L. & Genest, C. (1997b). Bivariate distributions with extreme value attractors. Proposé au *Journal of the American Statistical Association* en février 1997.

Deheuvels, P. (1979). La fonction de dépendance empirique et ses propriétés. Un test non paramétrique d'indépendance. *Bulletin de l'Académie royale de Belgique (Classe des Sciences).*

Deheuvels, P. (1984). Probabilistic aspects of multivariate extremes. *In Statistical Extremes and Applications.* J. Tiago de Oliveira (Ed.). Reidel, Dordrecht, 117–130.

Fisher, R. A. & Tippett, L. H. C. (1928). Limiting forms of the frequency distributions of the largest or smallest member of a sample. *Proceedings of the Cambridge Philosophical Society,* 180–190.

Galambos, J. (1987). *The Asymptotic Theory of Extreme Order Statistics.* 2nd ed., Kreir, Melbourne, FL.

Genest, C. & Rivest, L.-P. (1993). Statistical inference procedures for bivariate archimedean copulas. *Journal of the American Statistical Association,* 1034–1043.

Ghoudi, K., Khoudraji, A. & Rivest, L.-P. (1997). Propriétés statistiques des copules de valeurs extrêmes bidimensionnelles. *La Revue Canadienne de Statistique, (sous-presse).*

Haan, L. de (1985). Extremes in Higher Dimensions: The Model and Some Statistics. *Proceedings of the 47th Session of the International Statistical Institute,* Amsterdam: Holland, 1–16.

Hildebrandt, T. H. (1963). *Introduction to the theory of integration.* Academic Press, New York.

Joe, H., Smith, R. L. & Weissman, I. (1992). Bivariate threshold methods for extremes. *Journal of the Royal Statistical Society, Series B,* **54**, 171–183.

Khoudraji, A. (1995). Contributions à l'étude des copules et à la modélisation de valeurs extrêmes bivariées. *Thèse de doctorat, Université Laval, Québec, Canada.*

Mises, R. von (1923). Uber die Variationsbreite einer Beobachtungsreihe. *Sitzungsberichte Berlin. Math Ges.* **22**, 3.

Mises, R. von (1954). La distribution de la plus grande de n valeurs. In *Selected Papers II.* Amer. Math. Soc., Providence, R.I., 271–294.

Pickands III, J. (1981). Multivariate extreme value distributions. *In Proc. 43rd Session I.S.I. (Buenos Aires).* 859–878.

Pickands III, J. (1975). Statistical inference using extreme order statistics. *Ann. Statist.,* **3** 119–131.

Resnick, S. I. (1987). *Extreme Values, Regular Variation and Point Processes.* Springer-Verlag, New York.

Sklar, A. (1959). Fonctions de répartition à n dimensions et leurs marges. *Publications de l'Institut de statistique de l'Université de Paris,* **8**, 229–231.

Tawn, J. A. (1988). Bivariate extreme value theory-models and estimation. *Biometrika,* **75**, 397–415.

Tiago de Oliveira, J. (1984). Bivariate models for extremes. J. Tiago de Oliveira (Ed.). Reidel, Dordrecht, 859–878.

Annexe A

Diagrammes en boîte pour l'analyse 1

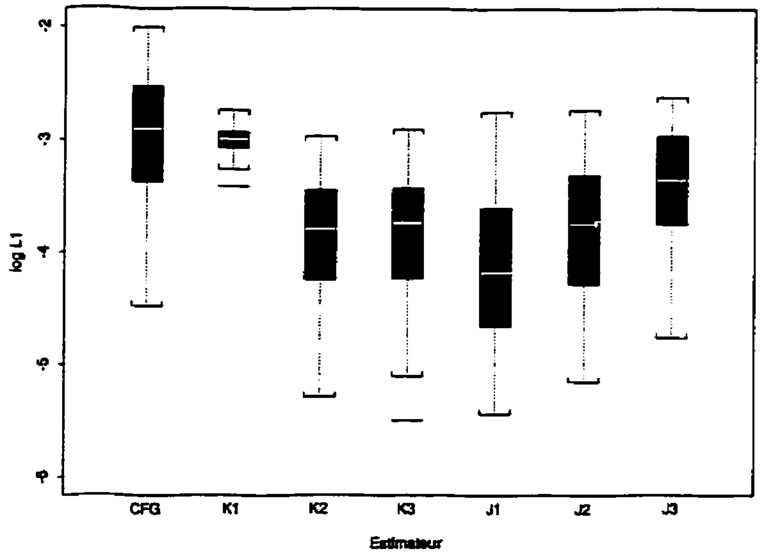


Figure A.1: *Diagrammes en boîte des erreurs* $log(L_1)$ *de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec* A_1^* *et* $\phi_{1,\alpha}$, $\alpha = 4/7$, *obtenus à partir de 100 échantillons de taille 2500.*

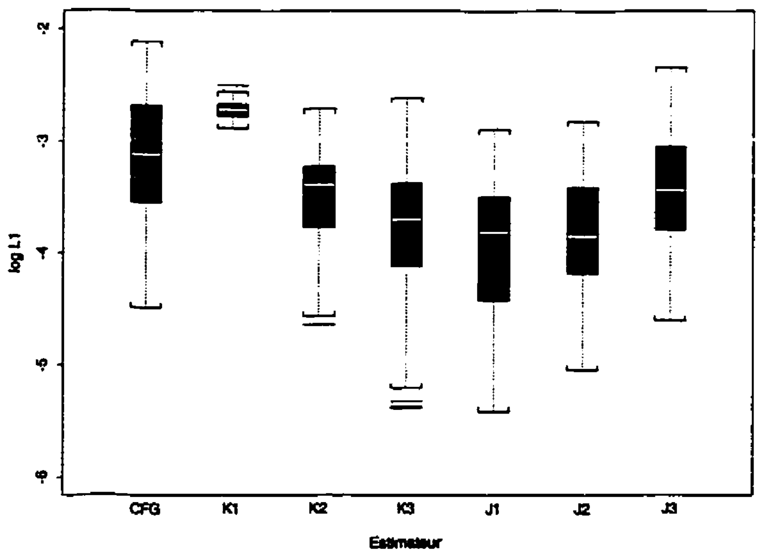


Figure A.2: *Diagrammes en boîte des erreurs* $log(L_1)$ *de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec* A_1^* *et* $\phi_{1,\alpha}$, $\alpha = 1.6$, *obtenus à partir de 100 échantillons de taille 2500.*

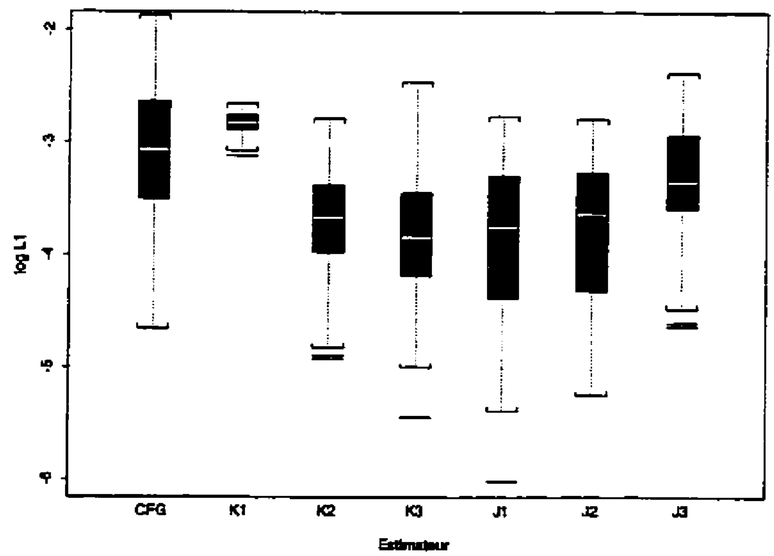


Figure A.3: *Diagrammes en boîte des erreurs $log(L_1)$ de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec A_1^* et $\phi_{2,\alpha}$, $\alpha = 2.09$, obtenus à partir de 100 échantillons de taille 2500.*

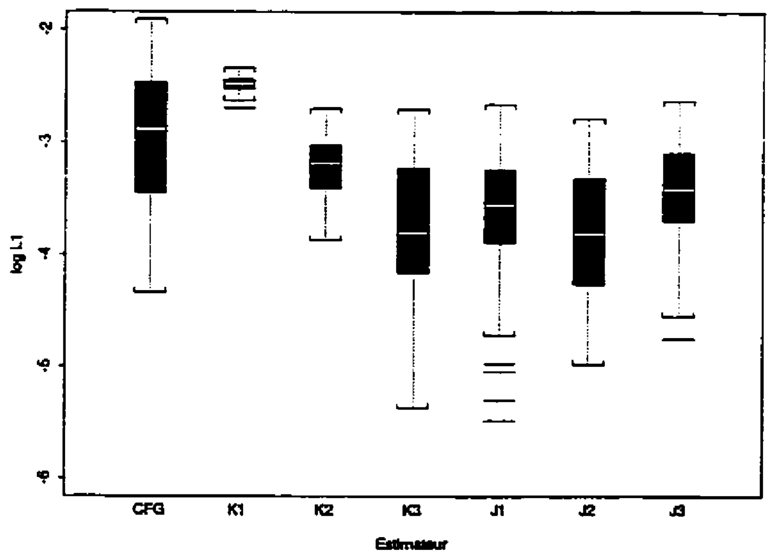


Figure A.4: *Diagrammes en boîte des erreurs $log(L_1)$ de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec A_1^* et $\phi_{2,\alpha}$, $\alpha = 4.8$, obtenus à partir de 100 échantillons de taille 2500.*

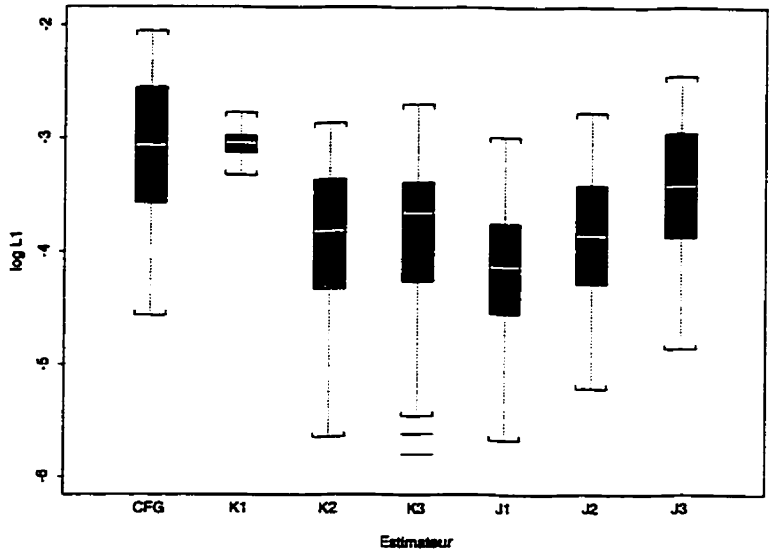


Figure A.5: *Diagrammes en boîte des erreurs log(L_1) de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec A_2^* et $\phi_{1,\alpha}$, $\alpha = 4/7$, obtenus à partir de 100 échantillons de taille 2500.*

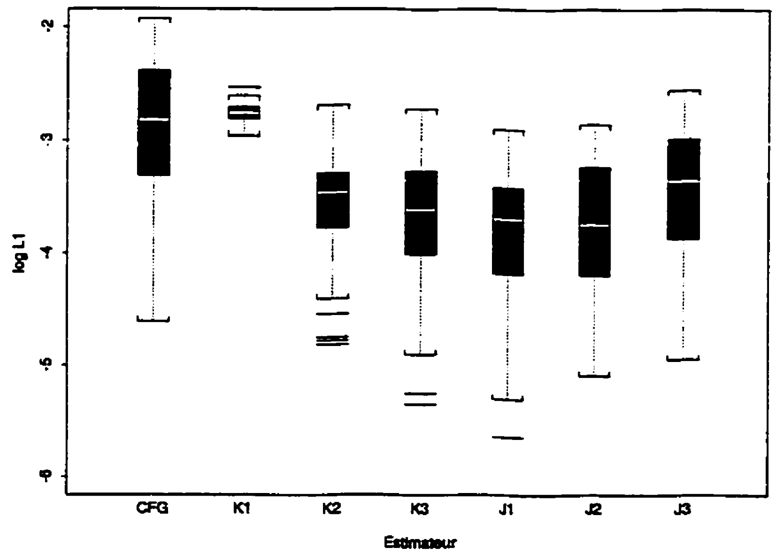


Figure A.6: *Diagrammes en boîte des erreurs log(L_1) de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec A_2^* et $\phi_{1,\alpha}$, $\alpha = 1.6$, obtenus à partir de 100 échantillons de taille 2500.*

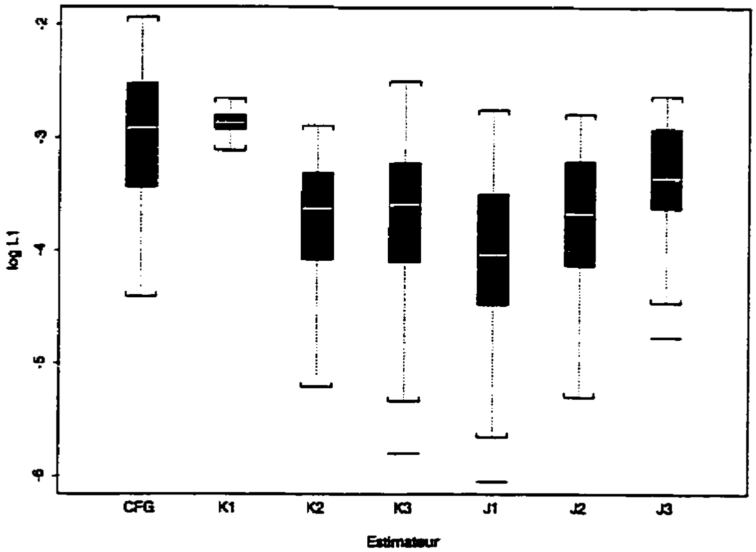


Figure A.7: *Diagrammes en boîte des erreurs* $log(L_1)$ *de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec* A_2^* *et* $\phi_{2,\alpha}$, $\alpha = 2.09$, *obtenus à partir de 100 échantillons de taille 2500.*

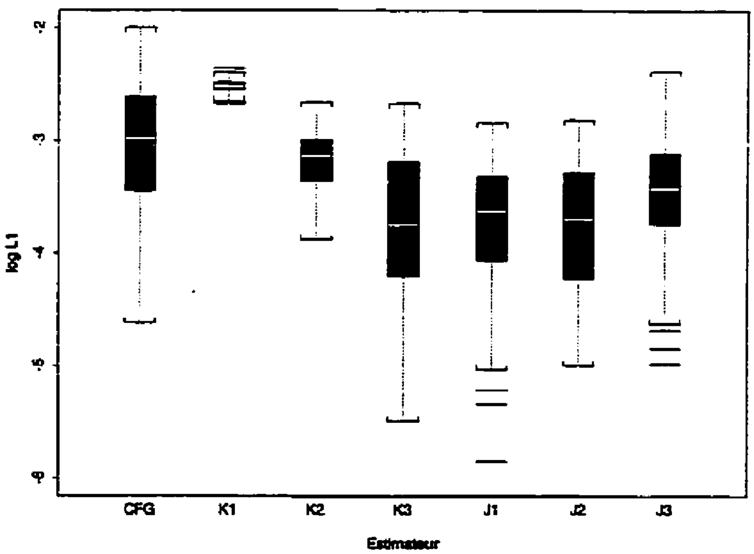


Figure A.8: *Diagrammes en boîte des erreurs* $log(L_1)$ *de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec* A_2^* *et* $\phi_{2,\alpha}$, $\alpha = 4.8$, *obtenus à partir de 100 échantillons de taille 2500.*

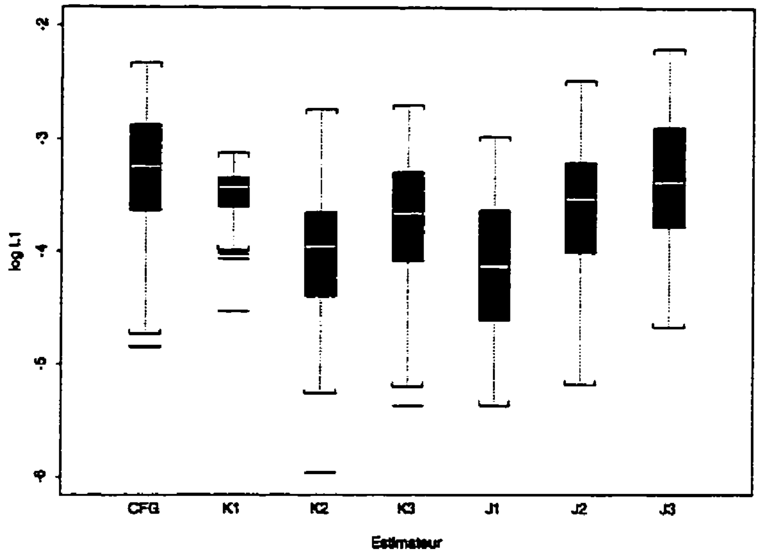


Figure A.9: *Diagrammes en boîte des erreurs* $log(L_1)$ *de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec* A_3^* *et* $\phi_{1,\alpha}$, $\alpha = 0.972$, *obtenus à partir de 100 échantillons de taille 2500.*

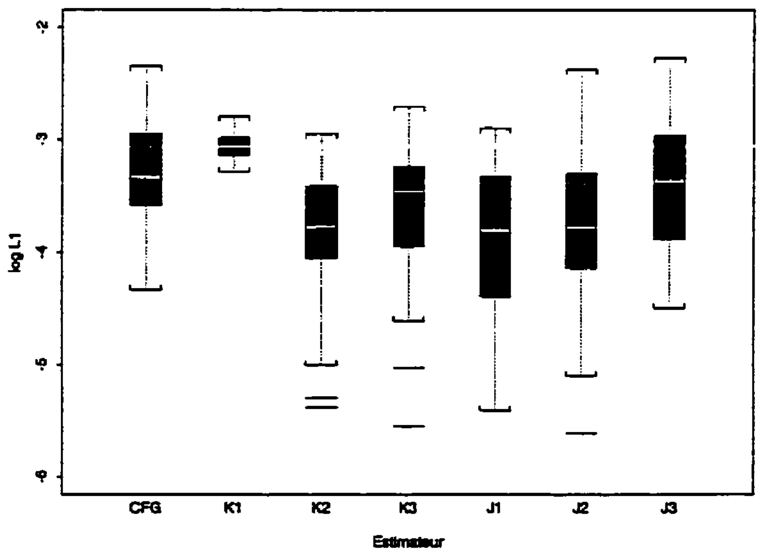


Figure A.10: *Diagrammes en boîte des erreurs* $log(L_1)$ *de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec* A_3^* *et* $\phi_{1,\alpha}$, $\alpha = 2.953$, *obtenus à partir de 100 échantillons de taille 2500.*

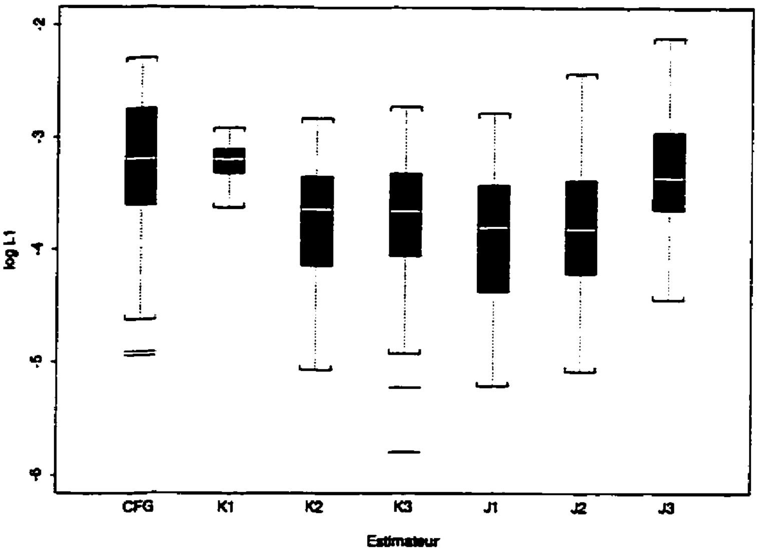


Figure A.11: *Diagrammes en boîte des erreurs log(L_1) de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec A_3^* et $\phi_{2,\alpha}$, $\alpha = 3.23$, obtenus à partir de 100 échantillons de taille 2500.*

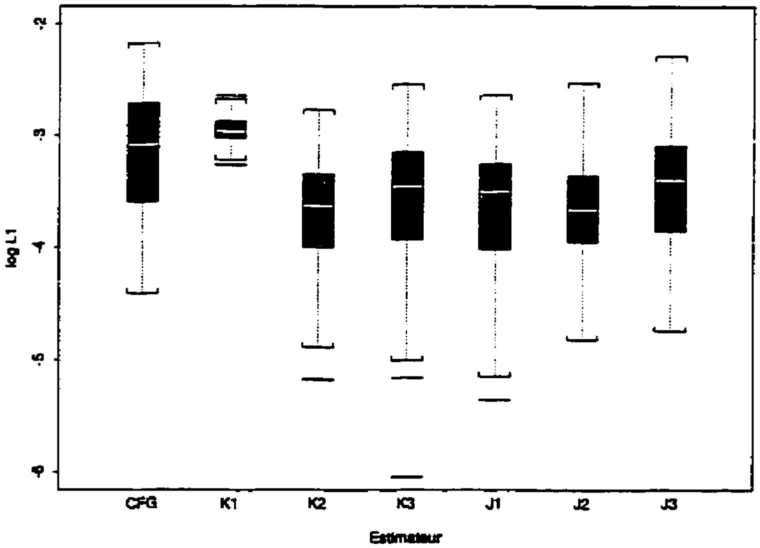


Figure A.12: *Diagrammes en boîte des erreurs log(L_1) de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec A_3^* et $\phi_{2,\alpha}$, $\alpha = 7.83$, obtenus à partir de 100 échantillons de taille 2500.*

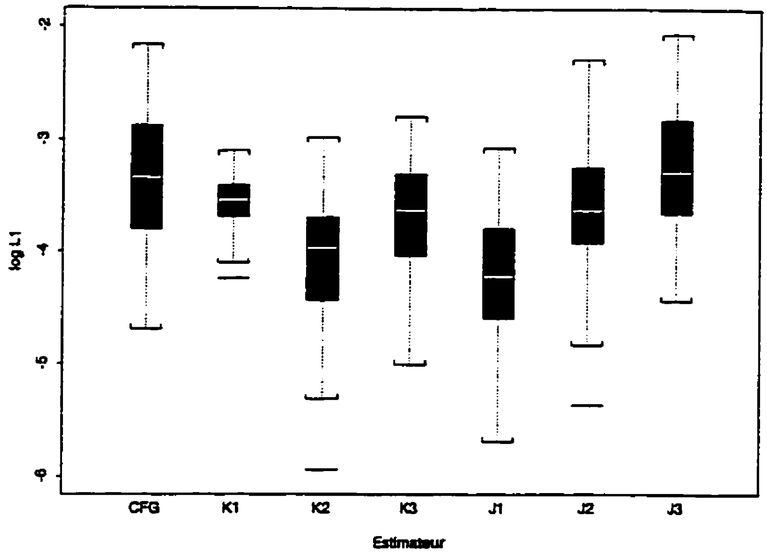


Figure A.13: *Diagrammes en boîte des erreurs log(L_1) de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec A_4^* et $\phi_{1,\alpha}$, $\alpha = 0.972$, obtenus à partir de 100 échantillons de taille 2500.*

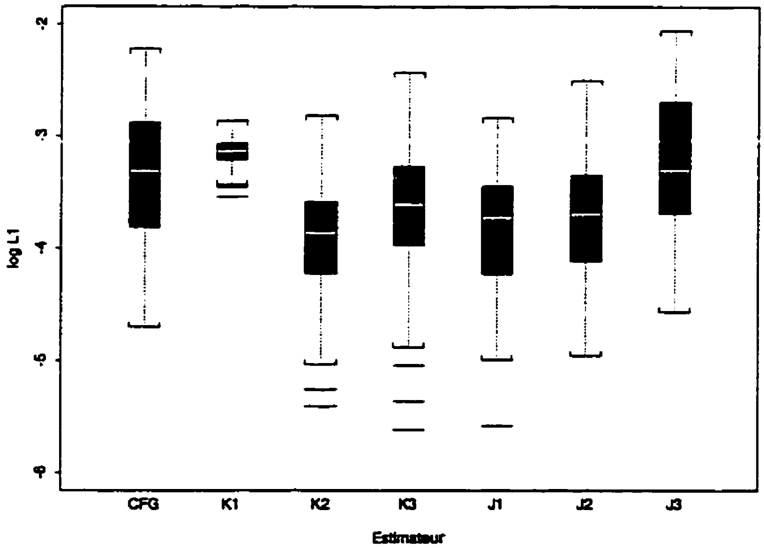


Figure A.14: *Diagrammes en boîte des erreurs log(L_1) de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec A_4^* et $\phi_{1,\alpha}$, $\alpha = 2.953$, obtenus à partir de 100 échantillons de taille 2500.*

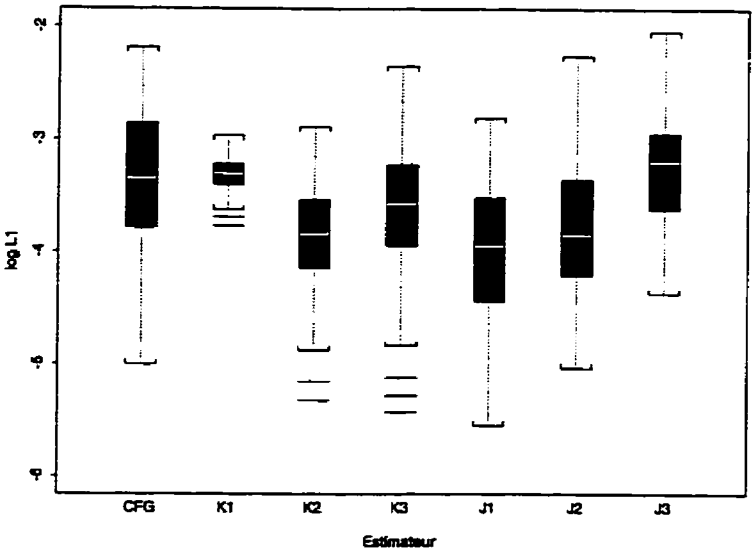


Figure A.15: *Diagrammes en boîte des erreurs* $log(L_1)$ *de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec* A_4^* *et* $\phi_{2,\alpha}$, $\alpha = 3.23$, *obtenus à partir de 100 échantillons de taille 2500.*

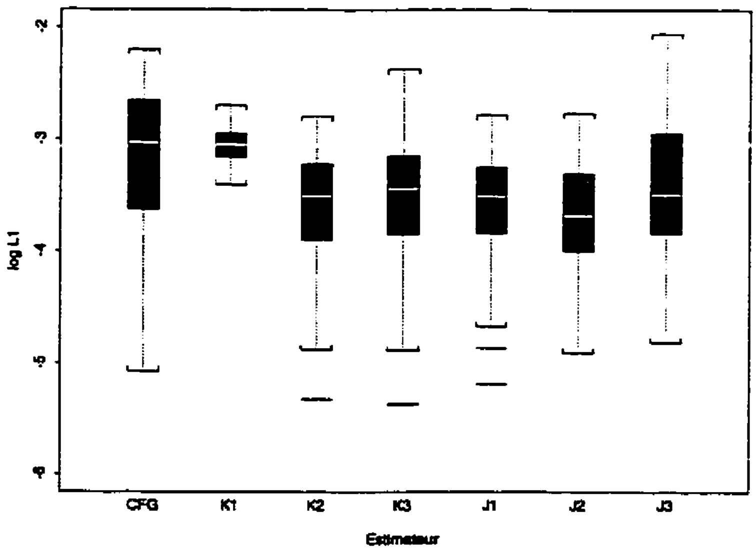


Figure A.16: *Diagrammes en boîte des erreurs* $log(L_1)$ *de 7 estimateurs de la fonction de dépendance d'une copule Archimax avec* A_4^* *et* $\phi_{2,\alpha}$, $\alpha = 7.83$, *obtenus à partir de 100 échantillons de taille 2500.*

Annexe B

Graphiques des estimations de la fonction de dépendance pour l'analyse 1

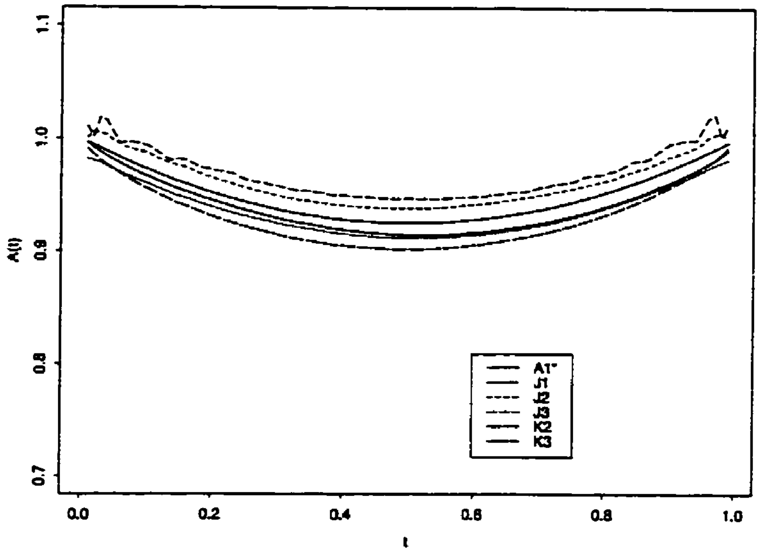


Figure B.1: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Clayton avec $\alpha = 4/7$, et où A^* est la fonction A_1^*, avec $\tau_{A^*} = 0.10$.*

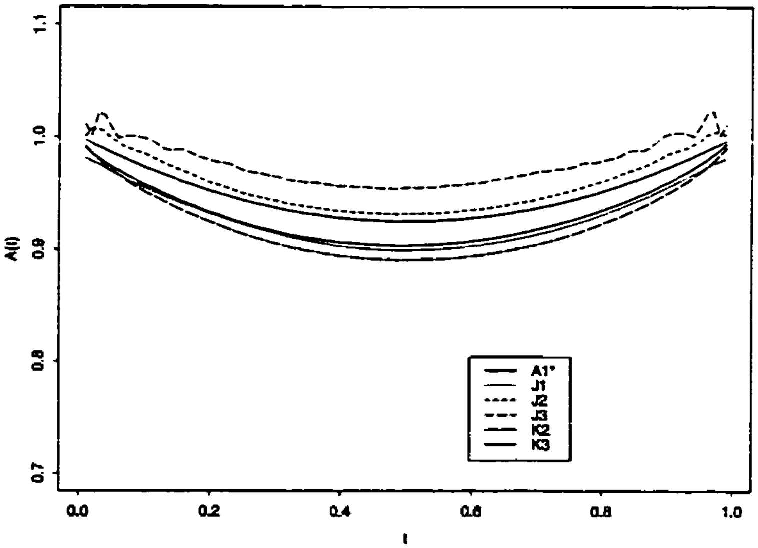


Figure B.2: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Clayton avec $\alpha = 1.6$, et où A^* est la fonction A_1^*, avec $\tau_{A^*} = 0.10$.*

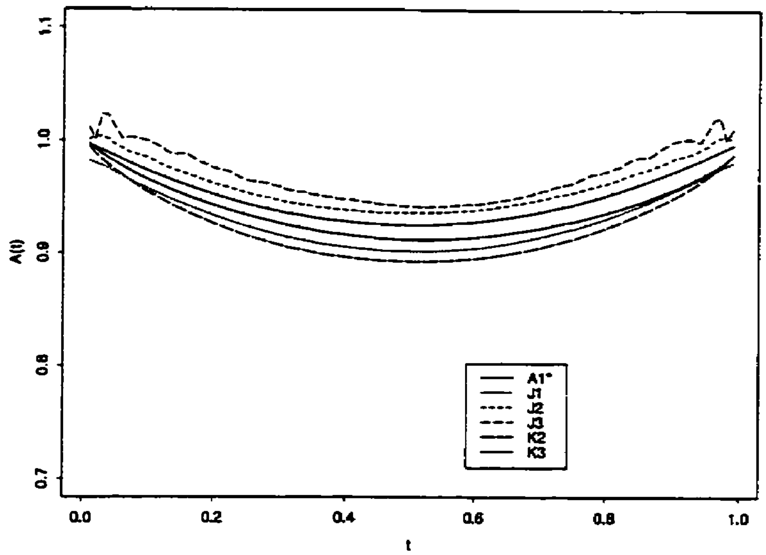


Figure B.3: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Franck avec $\alpha = 2.09$, et où A^* est la fonction A_1^*, avec $\tau_{A^*} = 0.10$.*

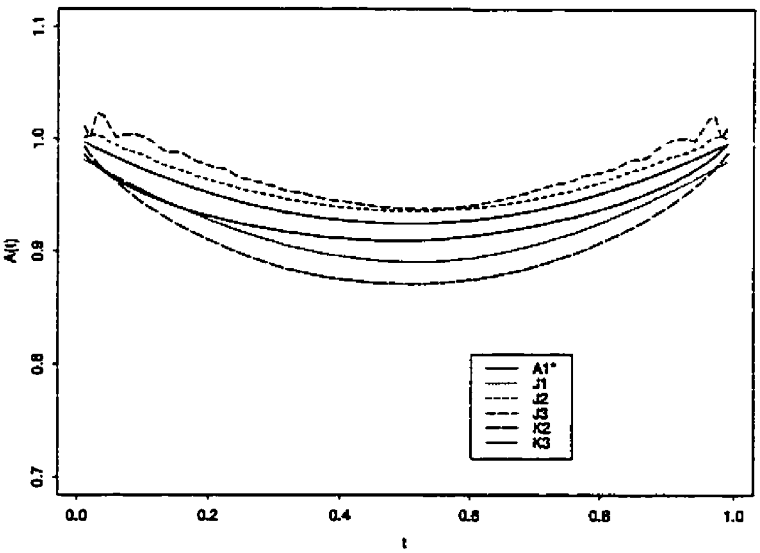


Figure B.4: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Franck avec $\alpha = 4.8$, et où A^* est la fonction A_1^*, avec $\tau_{A^*} = 0.10$.*

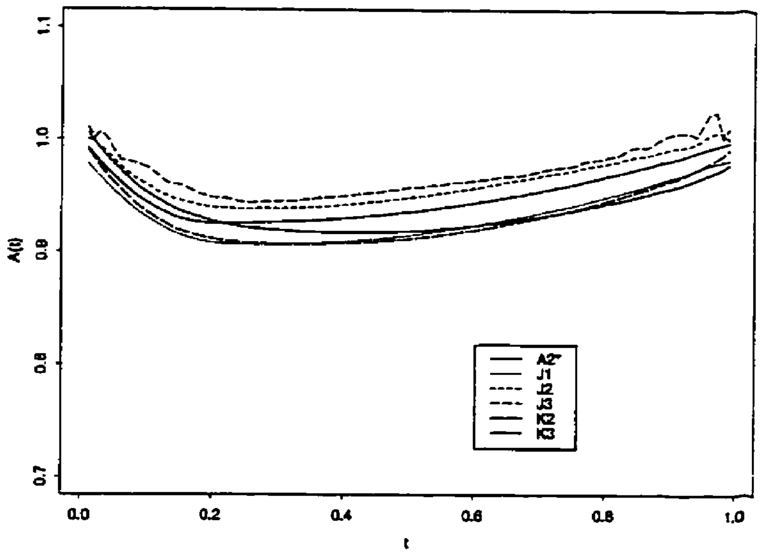


Figure B.5: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Clayton avec $\alpha = 4/7$, et où A^* est la fonction A_2^*, avec $\tau_{A^*} = 0.10$.*

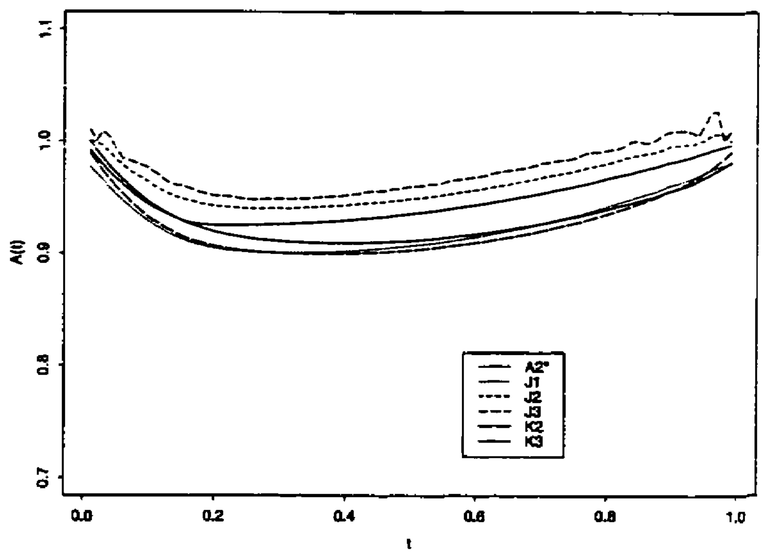


Figure B.6: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Clayton avec $\alpha = 1.6$, et où A^* est la fonction A_2^*, avec $\tau_{A^*} = 0.10$.*

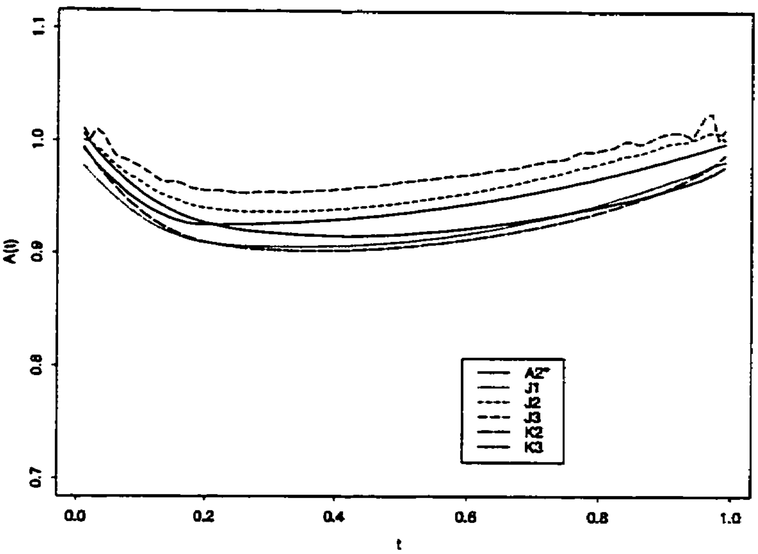


Figure B.7: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Franck avec $\alpha = 2.09$, et où A^* est la fonction A_2^*, avec $\tau_{A^*} = 0.10$.*

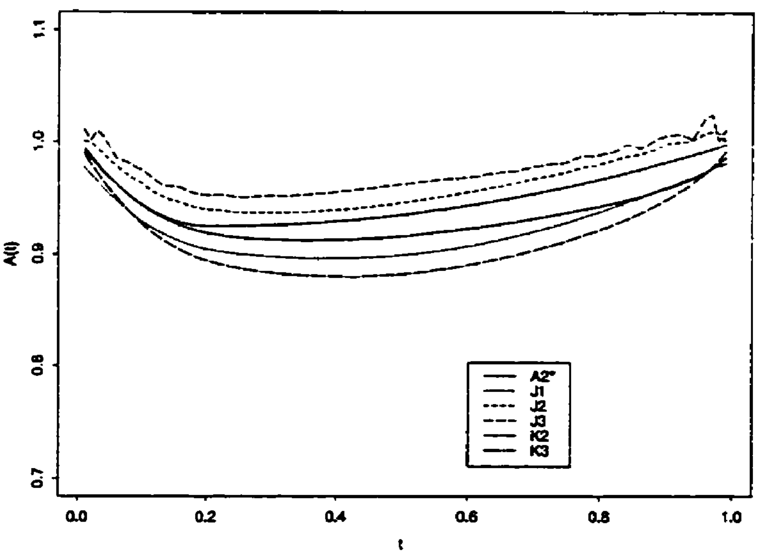


Figure B.8: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Franck avec $\alpha = 4.8$, et où A^* est la fonction A_2^*, avec $\tau_{A^*} = 0.10$.*

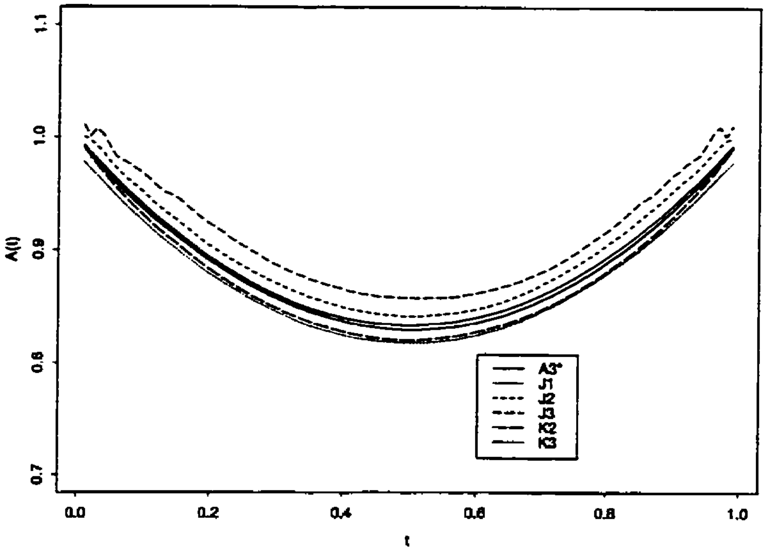


Figure B.9: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Clayton avec* $\alpha = 0.972$, *et où A^* est la fonction A_3^*, avec* $\tau_{A^*} = 0.26$.

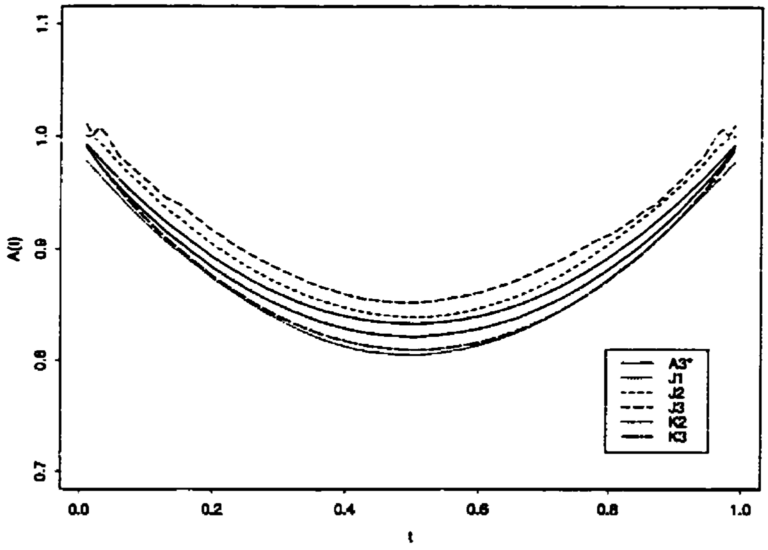


Figure B.10: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Clayton avec* $\alpha = 2.953$, *et où A^* est la fonction A_3^*, avec* $\tau_{A^*} = 0.26$.

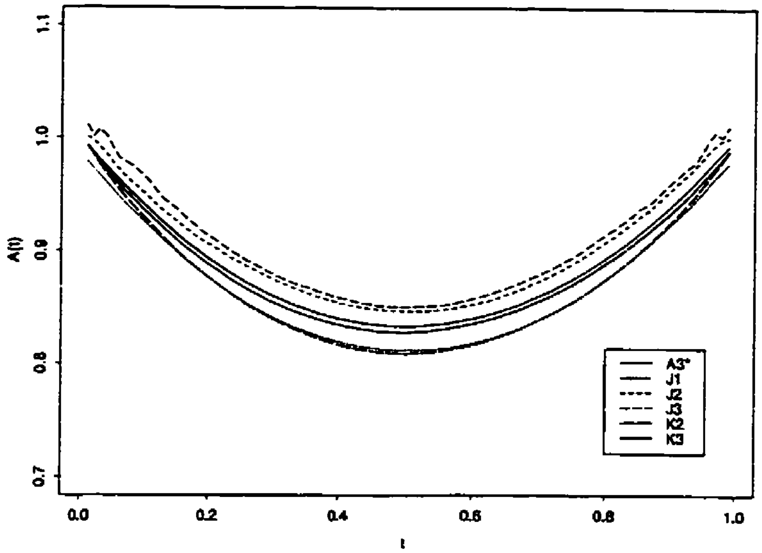


Figure B.11: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur* ϕ *est celui de Franck avec* $\alpha = 3.23$*, et où* A^* *est la fonction* A_3^**, avec* $\tau_{A^*} = 0.26$.

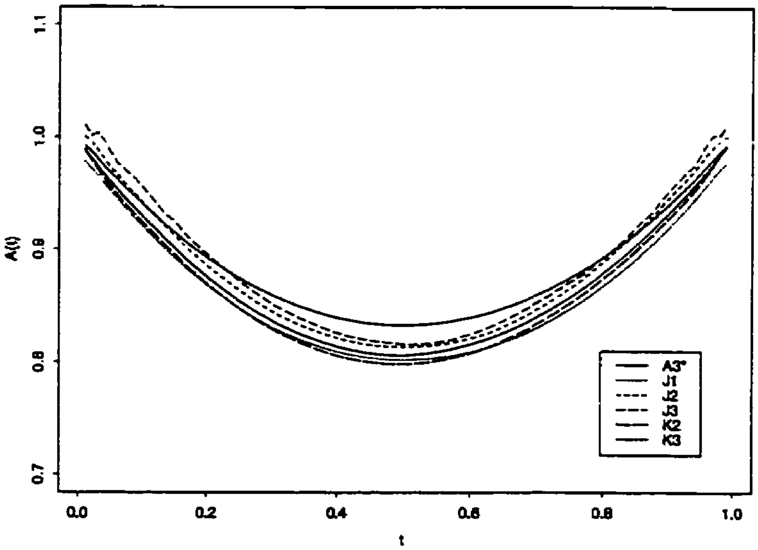


Figure B.12: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur* ϕ *est celui de Franck avec* $\alpha = 7.83$*, et où* A^* *est la fonction* A_3^**, avec* $\tau_{A^*} = 0.26$.

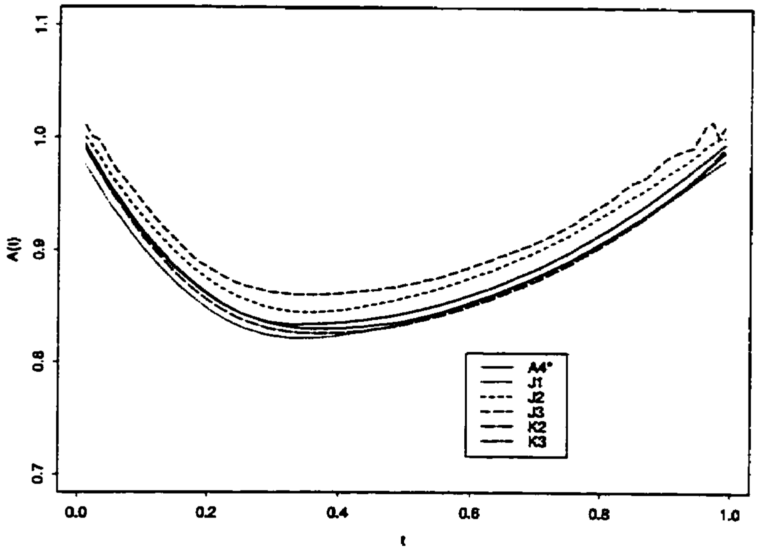


Figure B.13: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Clayton avec* $\alpha = 0.972$, *et où* A^* *est la fonction* A_4^*, *avec* $\tau_{A^*} = 0.26$.

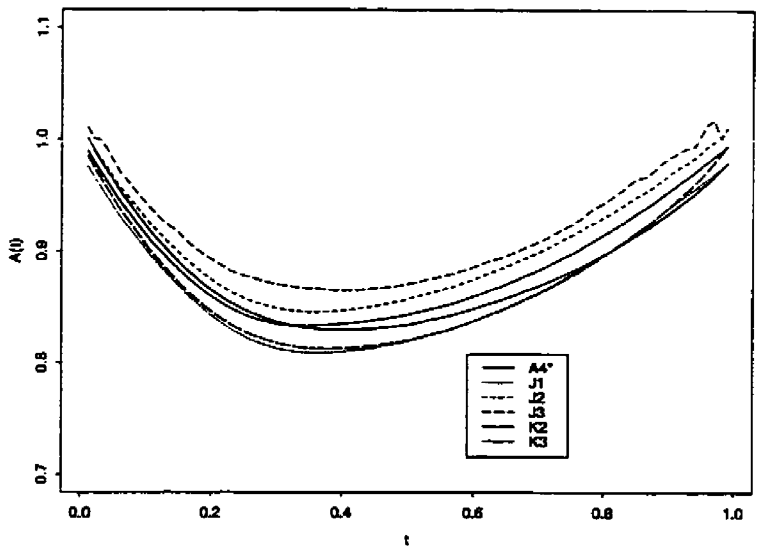


Figure B.14: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Clayton avec* $\alpha = 2.953$, *et où* A^* *est la fonction* A_4^*, *avec* $\tau_{A^*} = 0.26$.

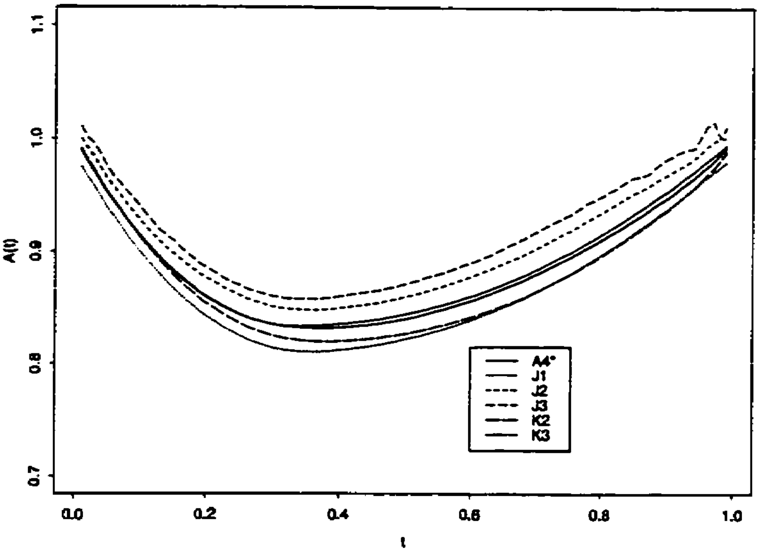


Figure B.15: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Franck avec* $\alpha = 3.23$*, et où A^* est la fonction A_4^*, avec* $\tau_{A^*} = 0.26$.

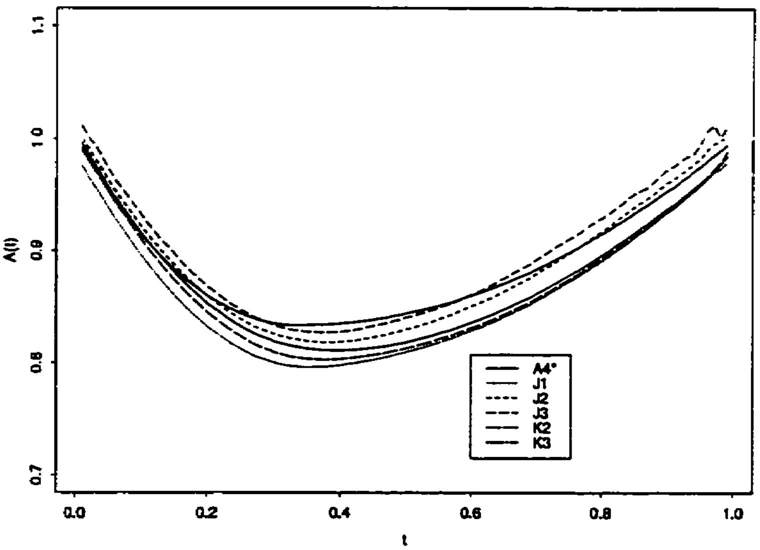


Figure B.16: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 2500 d'une copule Archimax, où le générateur ϕ est celui de Franck avec* $\alpha = 7.83$*, et où A^* est la fonction A_4^*, avec* $\tau_{A^*} = 0.26$.

Annexe C

Diagrammes en boîte pour l'analyse 2

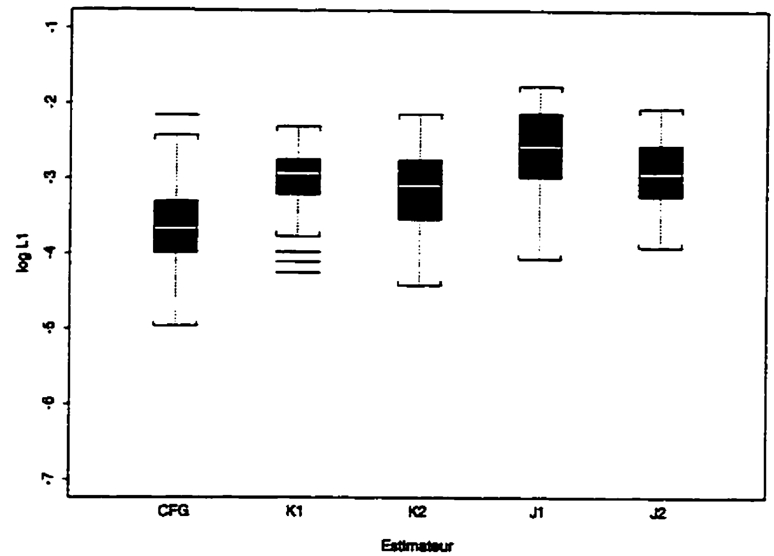


Figure C.1: *Diagrammes en boîte des erreurs log*(L_1) *de cinq estimateurs de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 50 d'une loi de valeurs extrêmes de Gumbel* A_r^**, symétrique par rapport à 1/2, où* $r = 4/3$ *avec* $\tau_{A^*} = 0.25$.

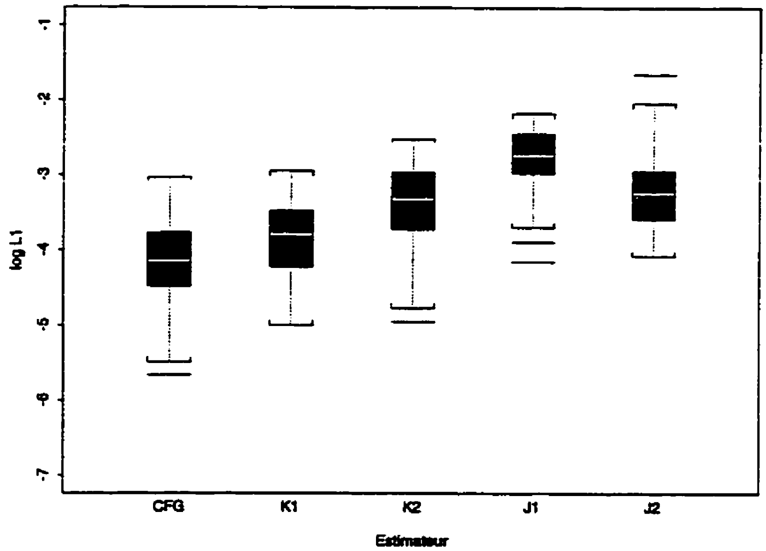


Figure C.2: *Diagrammes en boîte des erreurs log*(L_1) *de cinq estimateurs de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 50 d'une loi de valeurs extrêmes de Gumbel* A_r^**, symétrique par rapport à 1/2, où* $r = 2$ *avec* $\tau_{A^*} = 0.50$.

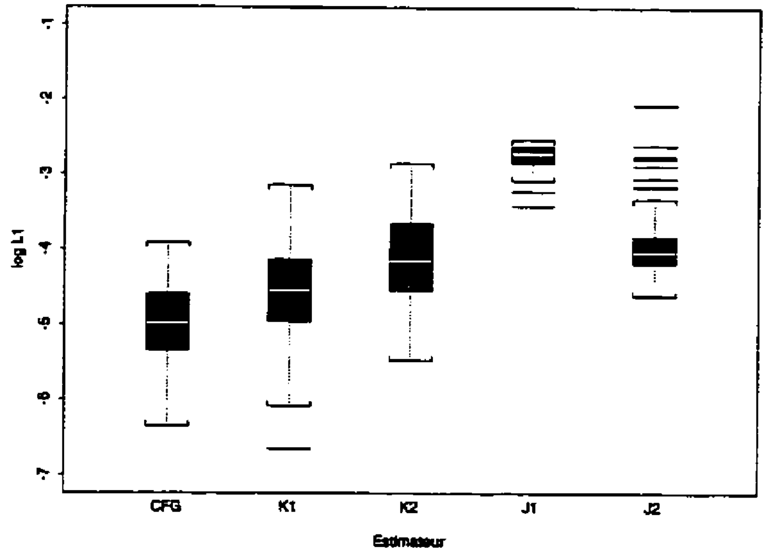


Figure C.3: *Diagrammes en boîte des erreurs* $log(L_1)$ *de cinq estimateurs de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 50 d'une loi de valeurs extrêmes de Gumbel* A_r^*, *symétrique par rapport à 1/2, où* $r = 4$ *avec* $\tau_{A^*} = 0.75$.

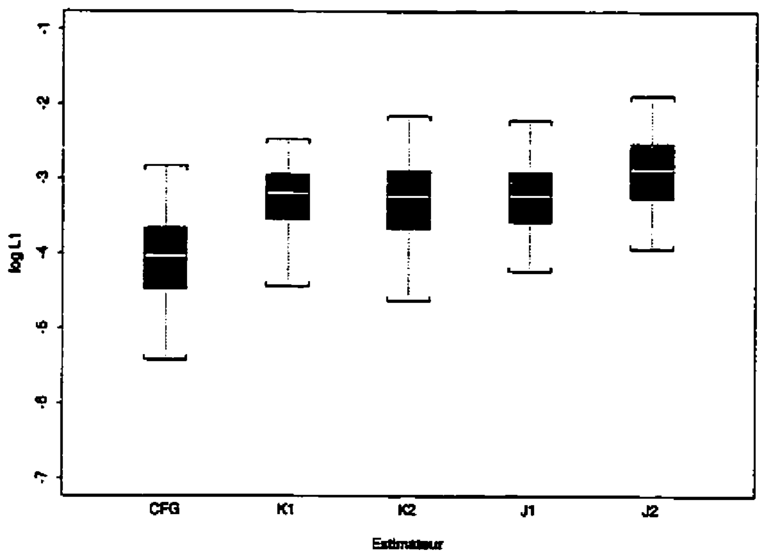


Figure C.4: *Diagrammes en boîte des erreurs* $log(L_1)$ *de cinq estimateurs de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 100 d'une loi de valeurs extrêmes de Gumbel* A_r^*, *symétrique par rapport à 1/2, où* $r = 4/3$ *avec* $\tau_{A^*} = 0.25$.

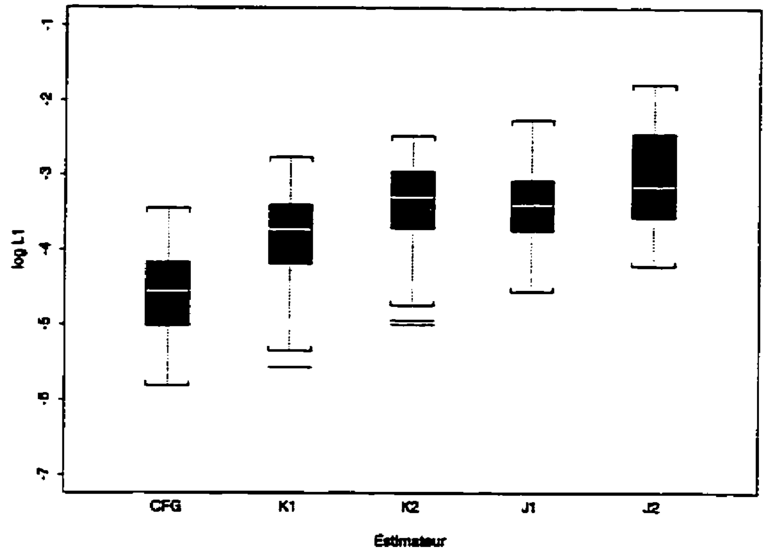


Figure C.5: *Diagrammes en boîte des erreurs* $log(L_1)$ *de cinq estimateurs de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 100 d'une loi de valeurs extrêmes de Gumbel* A_r^**, symétrique par rapport à 1/2, où* $r = 2$ *avec* $\tau_{A^*} = 0.50$.

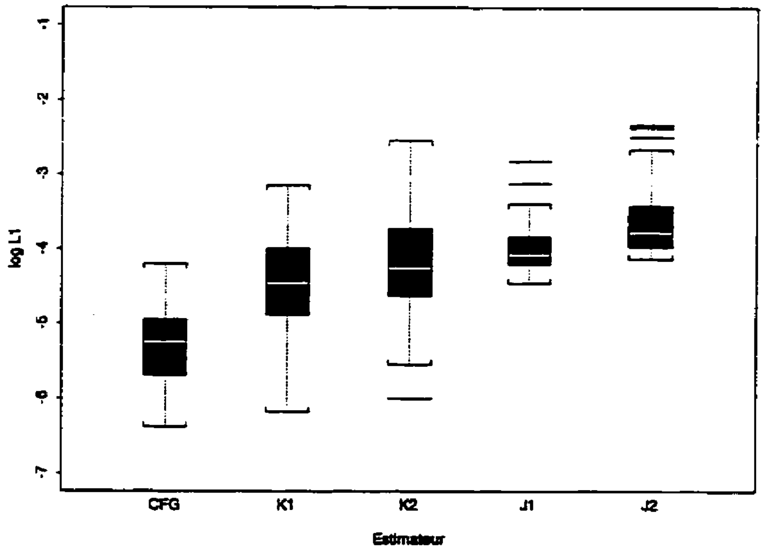


Figure C.6: *Diagrammes en boîte des erreurs* $log(L_1)$ *de cinq estimateurs de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 100 d'une loi de valeurs extrêmes de Gumbel* A_r^**, symétrique par rapport à 1/2, où* $r = 4$ *avec* $\tau_{A^*} = 0.75$.

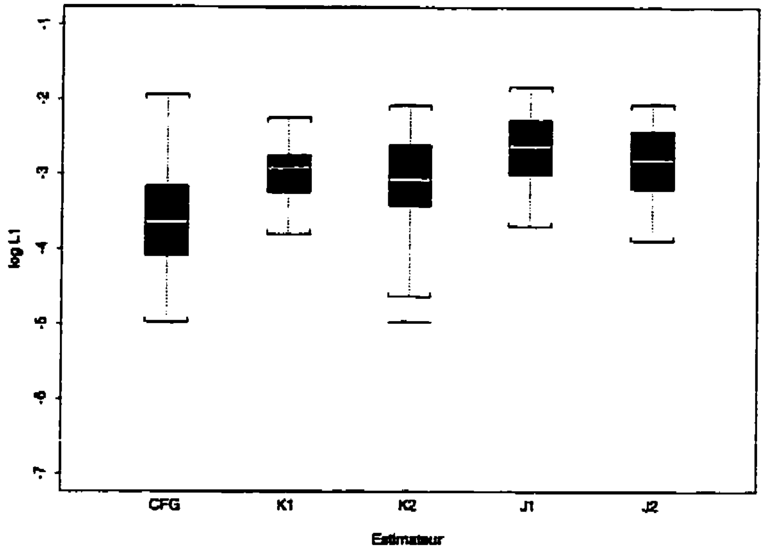


Figure C.7: *Diagrammes en boîte des erreurs* $log(L_1)$ *de cinq estimateurs de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 50 d'une loi de valeurs extrêmes asymétrique* $A^*_{\alpha,\beta,r}$, *où* $\alpha = 0.78$, $\beta = 0.97$ *et* $r = 1.42$, *avec* $\tau_{A^*} = 0.25$.

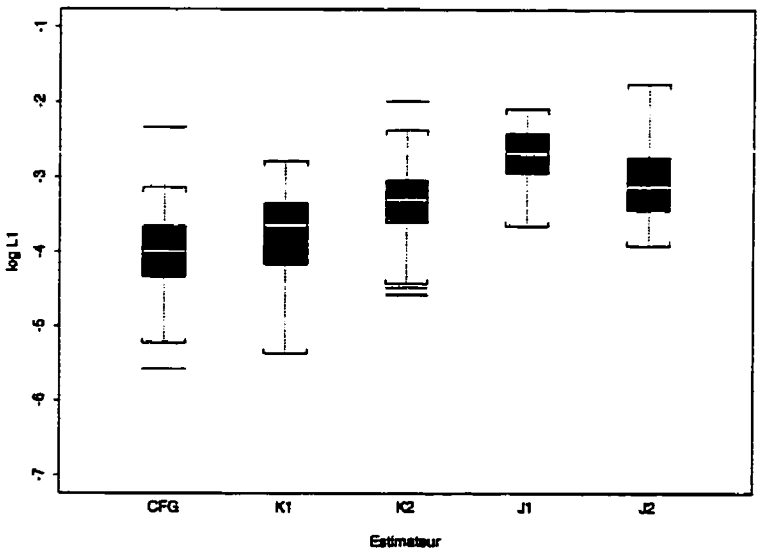


Figure C.8: *Diagrammes en boîte des erreurs* $log(L_1)$ *de cinq estimateurs de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 50 d'une loi de valeurs extrêmes asymétrique* $A^*_{\alpha,\beta,r}$, *où* $\alpha = 0.78$, $\beta = 0.97$ *et* $r = 2.58$, *avec* $\tau_{A^*} = 0.50$.

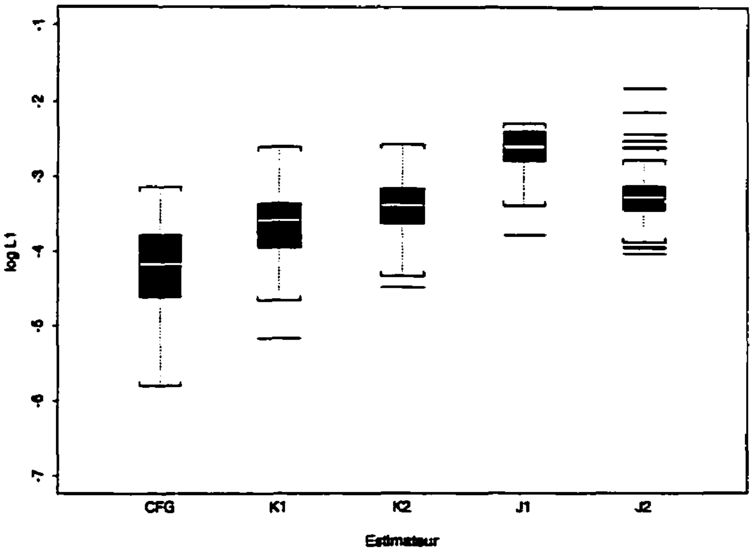


Figure C.9: *Diagrammes en boîte des erreurs* $log(L_1)$ *de cinq estimateurs de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 50 d'une loi de valeurs extrêmes asymétrique* $A^*_{\alpha,\beta,r}$, *où* $\alpha = 0.78$, $\beta = 0.97$ *et* $r = 50$, *avec* $\tau_{A^*} = 0.75$.

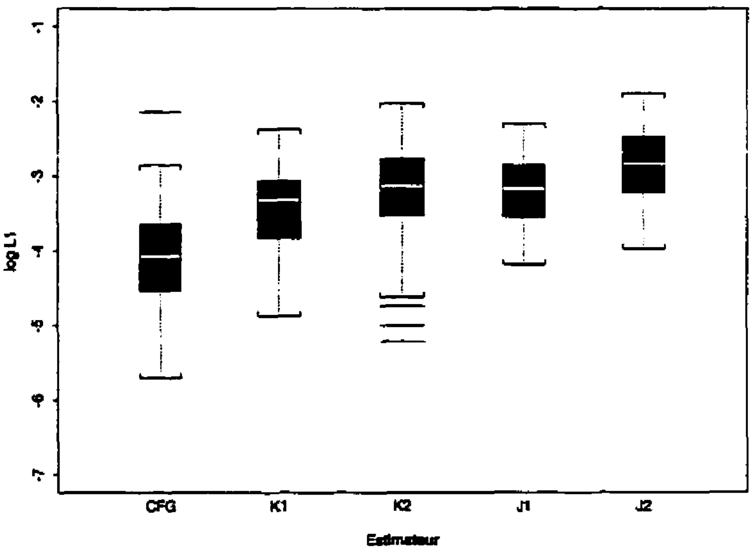


Figure C.10: *Diagrammes en boîte des erreurs* $log(L_1)$ *de cinq estimateurs de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 100 d'une loi de valeurs extrêmes asymétrique* $A^*_{\alpha,\beta,r}$, *où* $\alpha = 0.78$, $\beta = 0.97$ *et* $r = 1.42$, *avec* $\tau_{A^*} = 0.25$.

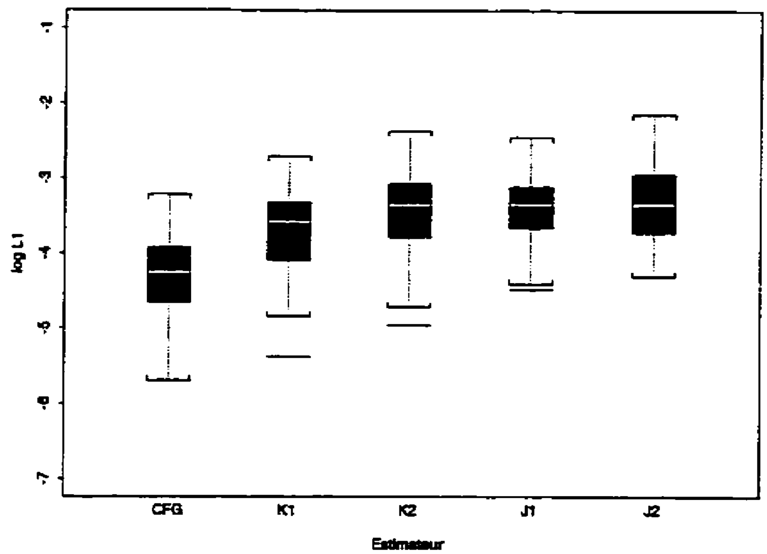


Figure C.11: *Diagrammes en boîte des erreurs* $log(L_1)$ *de cinq estimateurs de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 100 d'une loi de valeurs extrêmes asymétrique* $A^*_{\alpha,\beta,r}$, *où* $\alpha = 0.78$, $\beta = 0.97$ *et* $r = 2.58$, *avec* $\tau_{A^*} = 0.50$.

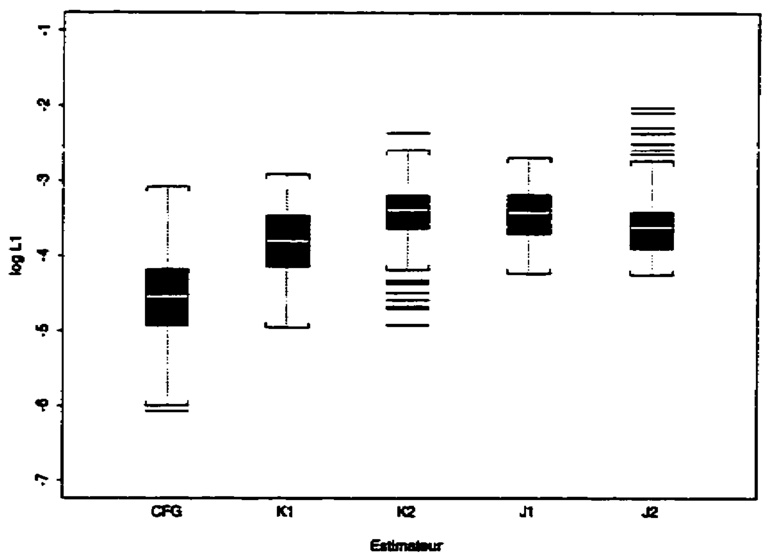


Figure C.12: *Diagrammes en boîte des erreurs* $log(L_1)$ *de cinq estimateurs de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 100 d'une loi de valeurs extrêmes asymétrique* $A^*_{\alpha,\beta,r}$, *où* $\alpha = 0.78$, $\beta = 0.97$ *et* $r = 50$, *avec* $\tau_{A^*} = 0.75$.

Annexe D

Graphiques des estimations de la fonction de dépendance pour l'analyse 2

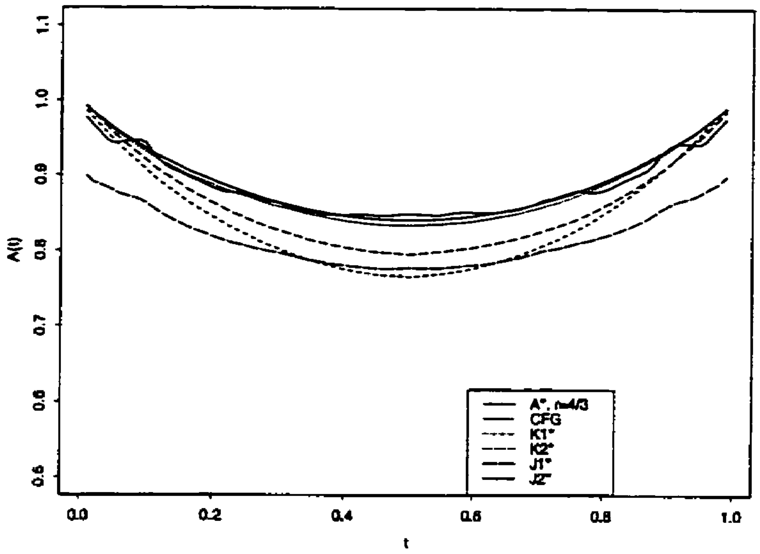


Figure D.1: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 50 d'une copule de valeurs extrêmes de Gumbel* A_r^*, *symétrique par rapport à 1/2, où* $r = 4/3$, *avec* $\tau_{A^*} = 0.25$.

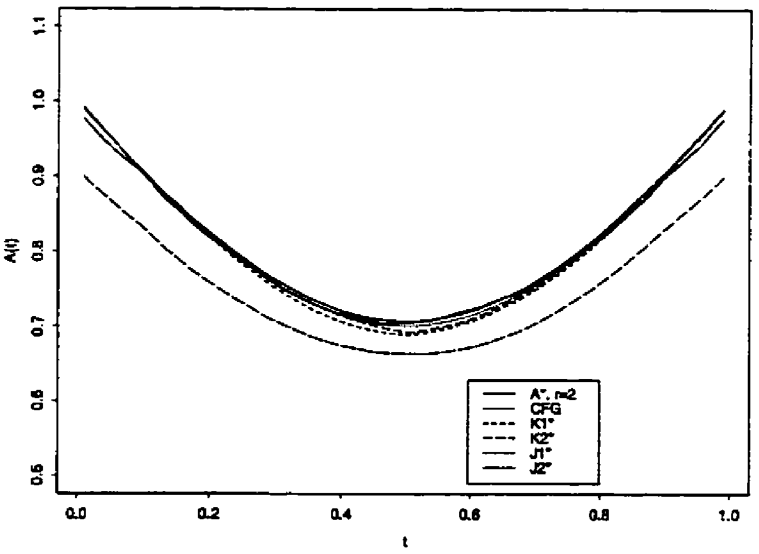


Figure D.2: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 50 d'une copule de valeurs extrêmes de Gumbel* A_r^*, *symétrique par rapport à 1/2, où* $r = 2$, *avec* $\tau_{A^*} = 0.50$.

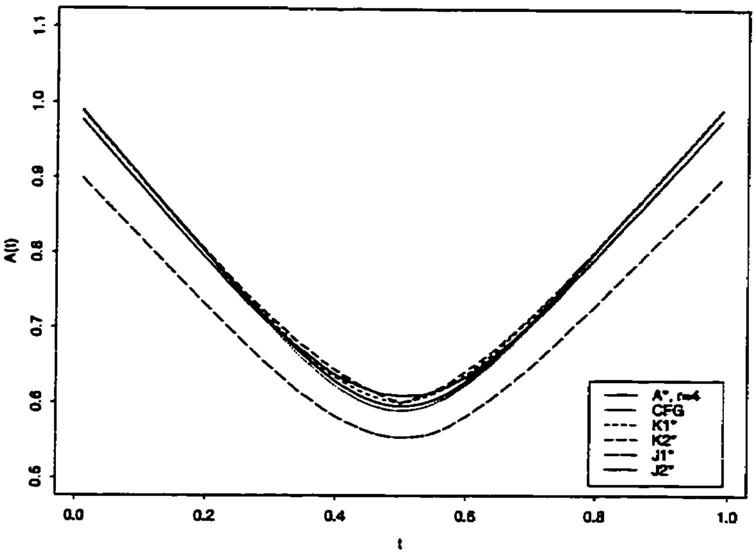


Figure D.3: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 50 d'une copule de valeurs extrêmes de Gumbel* A_r^*, *symétrique par rapport à 1/2, où* $r = 4$, *avec* $\tau_{A^*} = 0.75$.

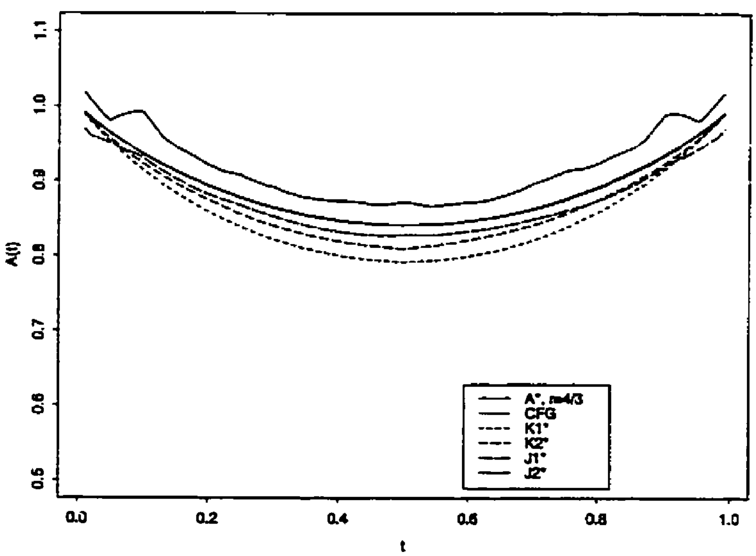


Figure D.4: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 100 d'une copule de valeurs extrêmes de Gumbel* A_r^*, *symétrique par rapport à 1/2, où* $r = 4/3$, *avec* $\tau_{A^*} = 0.25$.

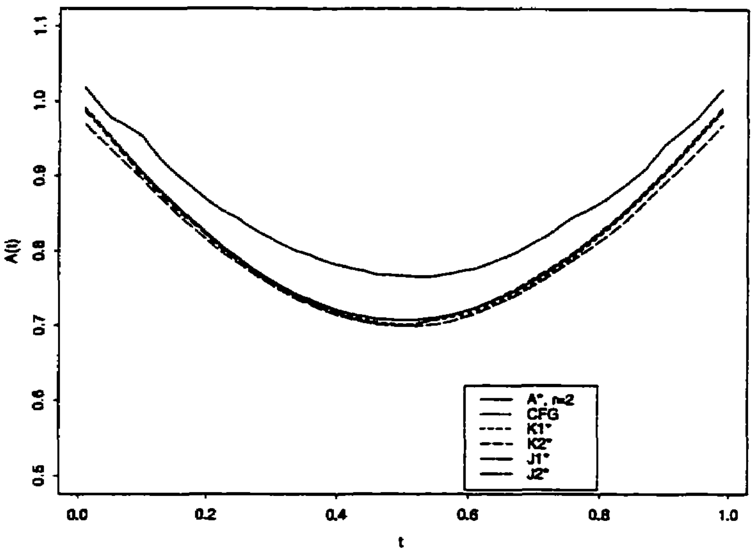


Figure D.5: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 100 d'une copule de valeurs extrêmes de Gumbel* A_r^**, symétrique par rapport à 1/2, où* $r = 2$*, avec* $\tau_{A^*} = 0.50$.

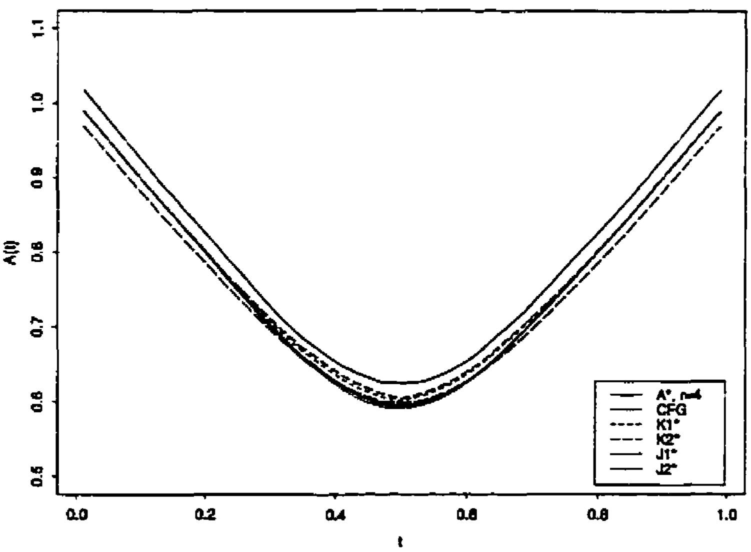


Figure D.6: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 100 d'une copule de valeurs extrêmes de Gumbel* A_r^**, symétrique par rapport à 1/2, où* $r = 4$*, avec* $\tau_{A^*} = 0.75$.

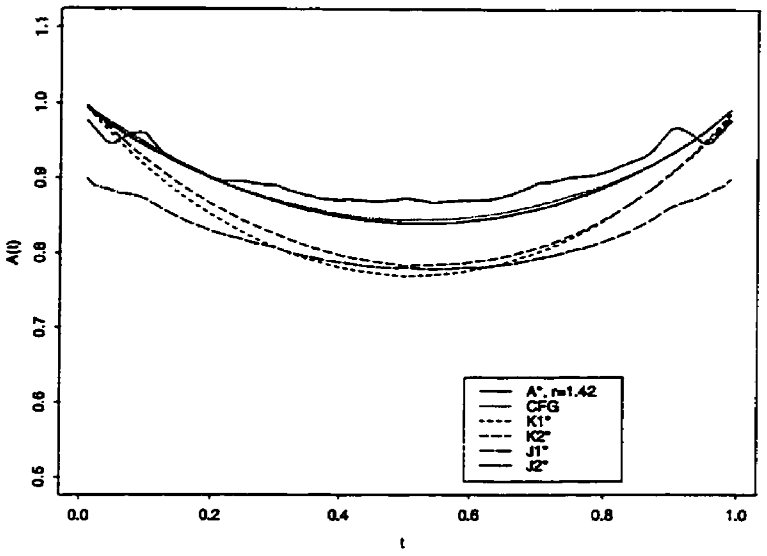


Figure D.7: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 50 d'une copule de valeurs extrêmes asymétrique* $A^*_{\alpha,\beta,r}$*, où* $\alpha = 0.78$, $\beta = 0.97$ *et* $r = 1.42$*, avec* $\tau_{A^*} = 0.25$.

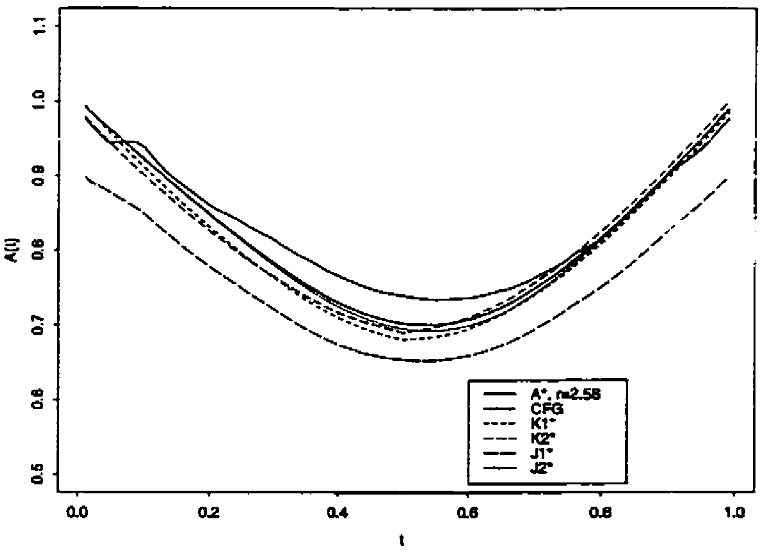


Figure D.8: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 50 d'une copule de valeurs extrêmes asymétrique* $A^*_{\alpha,\beta,r}$*, où* $\alpha = 0.78$, $\beta = 0.97$ *et* $r = 2.58$*, avec* $\tau_{A^*} = 0.50$.

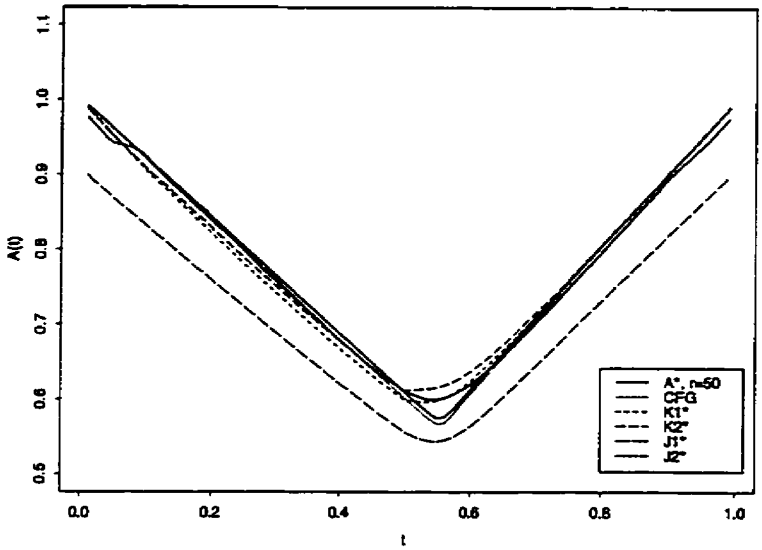


Figure D.9: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 50 d'une copule de valeurs extrêmes asymétrique* $A^*_{\alpha,\beta,r}$*, où* $\alpha = 0.78$, $\beta = 0.97$ *et* $r = 50$*, avec* $\tau_{A^*} = 0.75$.

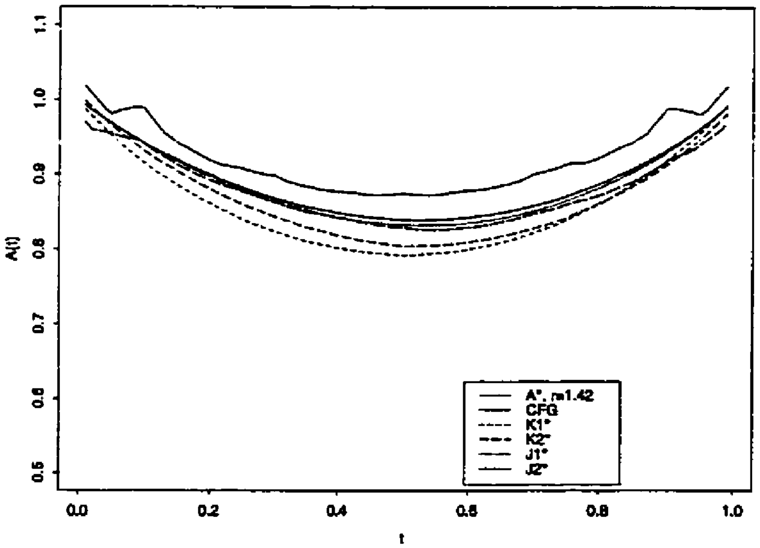


Figure D.10: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 100 d'une copule de valeurs extrêmes asymétrique* $A^*_{\alpha,\beta,r}$*, où* $\alpha = 0.78$, $\beta = 0.97$ *et* $r = 1.42$*, avec* $\tau_{A^*} = 0.25$.

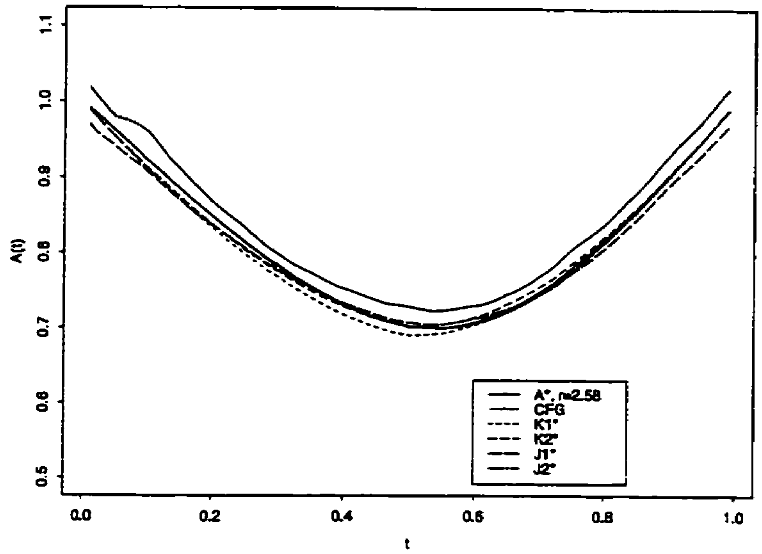


Figure D.11: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 100 d'une copule de valeurs extrêmes asymétrique* $A^*_{\alpha,\beta,r}$*, où* $\alpha = 0.78$, $\beta = 0.97$ *et* $r = 2.58$*, avec* $\tau_{A^*} = 0.50$.

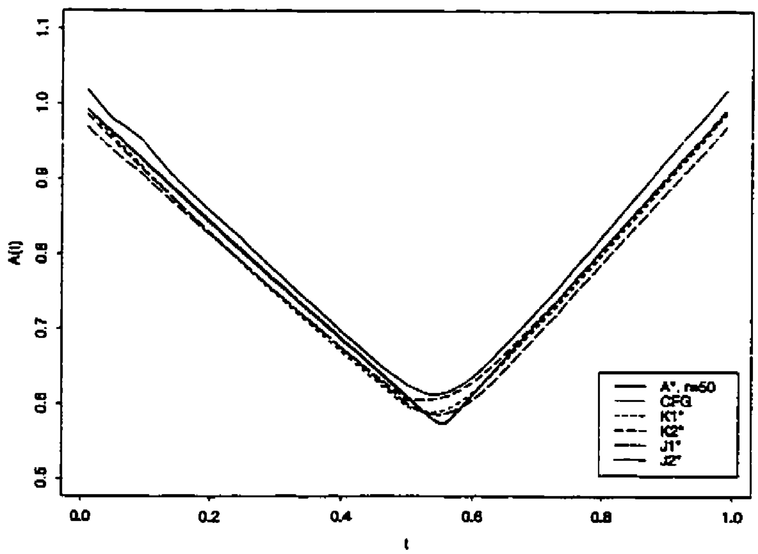
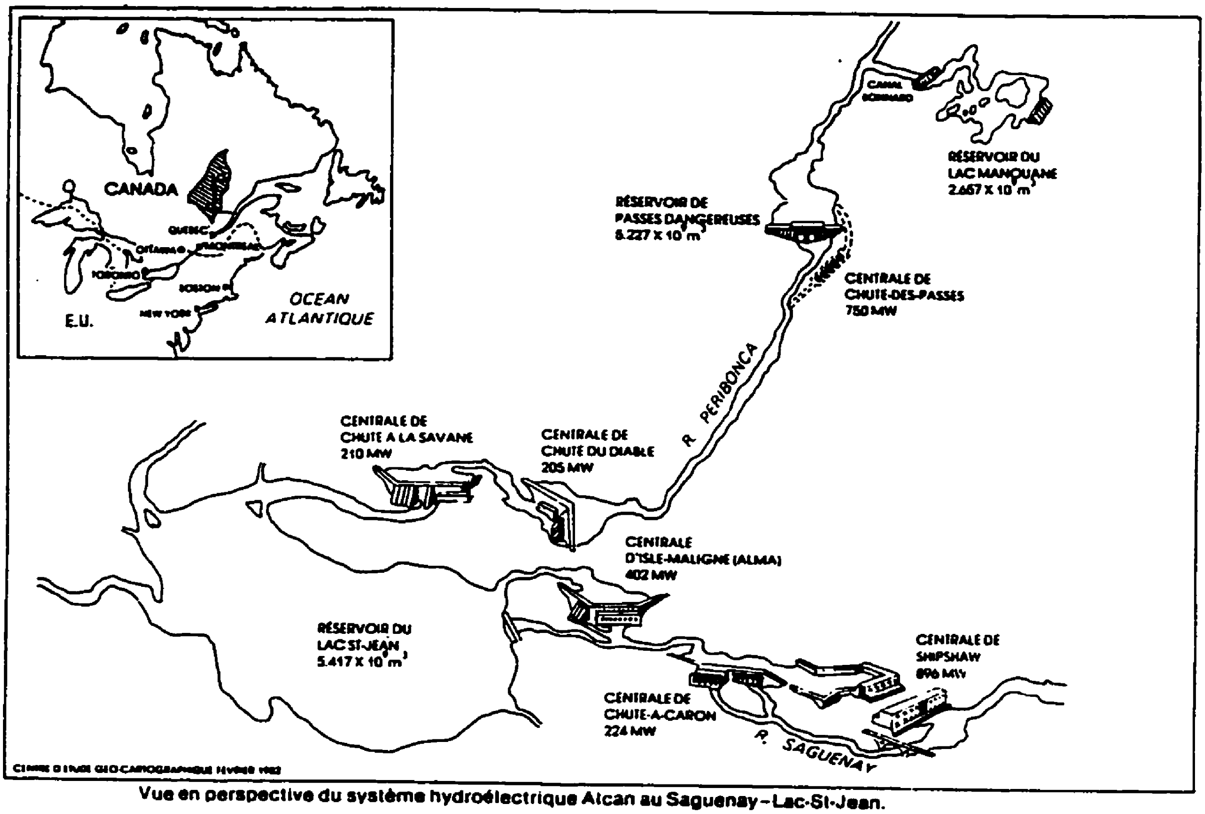


Figure D.12: *Estimation de la fonction de dépendance, obtenue à partir de 100 échantillons de taille 100 d'une copule de valeurs extrêmes asymétrique* $A^*_{\alpha,\beta,r}$*, où* $\alpha = 0.78$, $\beta = 0.97$ *et* $r = 50$*, avec* $\tau_{A^*} = 0.75$.

Annexe E

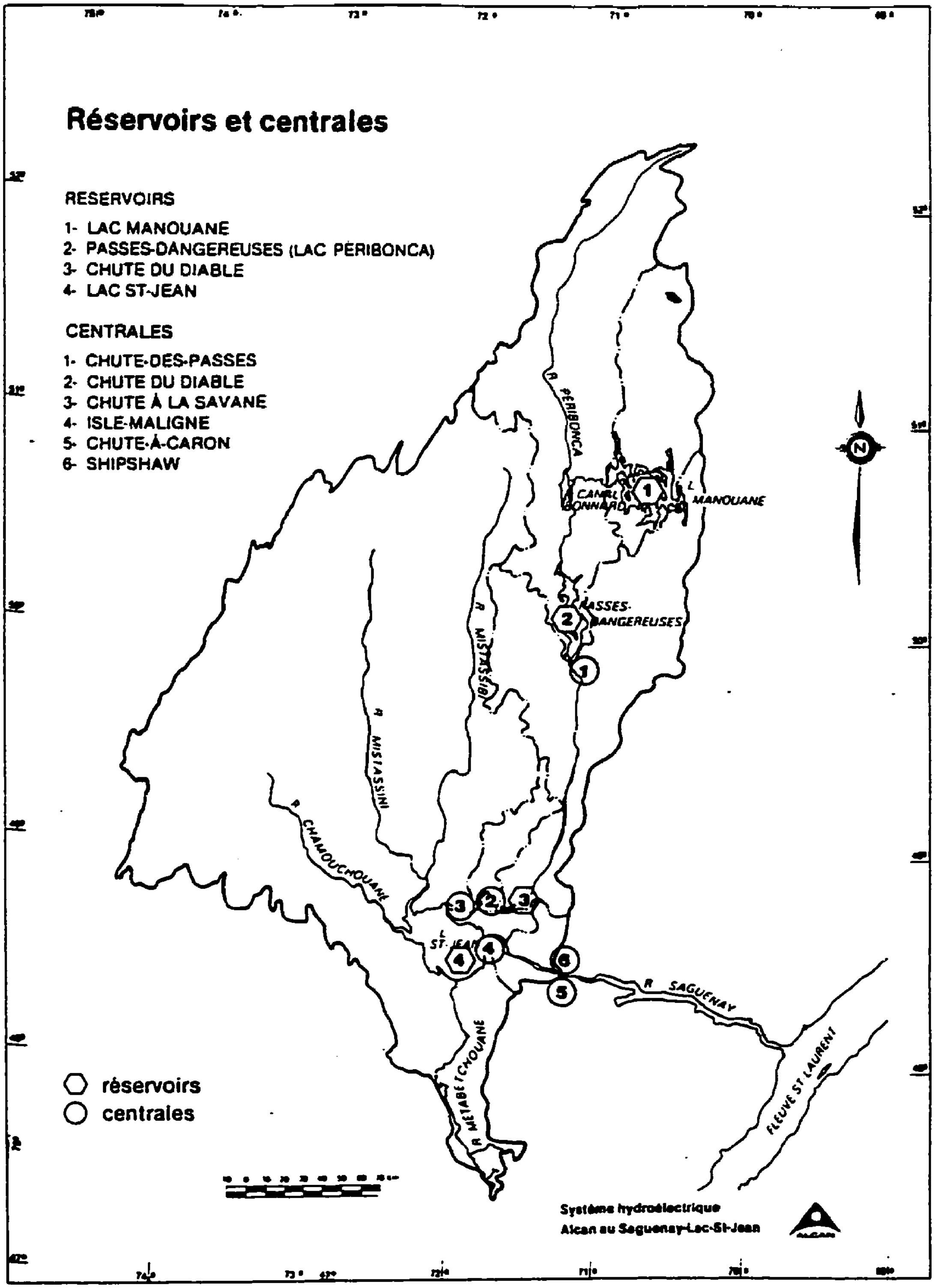
Cartes représentant les réservoirs Chute du Diable et Lac St-Jean

Carte 1



Vue en perspective du système hydroélectrique Alcan au Saguenay–Lac-St-Jean.

Carte 2


Réservoirs et centrales
RESERVOIRS
1- LAC MANOUANE
2- PASSES-DANGEREUSES (LAC PÉRIBONCA)
3- CHUTE DU DIABLE
4- LAC ST-JEAN
CENTRALES
1- CHUTE-DES-PASSES
2- CHUTE DU DIABLE
3- CHUTE À LA SAVANE
4- ISLE-MALIGNE
5- CHUTE-À-CARON
6- SHIPSHAW
R. PÉRIBONCA
CANAL BONNARD
L. MANOUANE
PASSES-DANGEREUSES
R. MISTASSIBI
R. MISTASSINI
R. CHAMOUCHOUANE
L. ST-JEAN
R. SAGUENAY
R. MÉTABETCHOUANE
FLEUVE ST LAURENT
réservoirs
centrales
Système hydroélectrique
Alcan au Saguenay-Lac-St-Jean

Annexe F

Programmes informatiques de fortran 77 utilisés pour l'analyse 1

```
***************************************************************************
PROGRAMME PRINCIPAL CALCULANT LES VALEURS DU TAU DE KENDALL ET DE L'ERREUR L1
 POUR DES ESTIMATIONS DE LA FONCTION A PAR LES METHODES DE JOE ET AL., DE
 KHOUDRAJI, ET DE CAPERAA ET AL., A PARTIR D'ECHANTILLONS DE TAILLE 2500 DE
 COPULE ARCHIMAX, OU LE TAU DE KENDALL THEORIQUE DE LA FONCTION A EST 0.1
***************************************************************************
        program analyseA1
        include 'common'
        integer q
        real*8 taustar
        parameter (q=100,taustar=0.1d0)
        integer nbobs,j,ind,l
        real*8 depart, xvec(n),yvec(n),taun,tau(d,d),erreurL1(d)
        real*8 Astarn(d),Astarn1(d,d),erreurL11(d,d)
        real*8 EQM(d),errLiber(d),errLicar(d),taubar(d)
        depart=drand(1212)
        open(1,file='sumal')
        open(2,file='resAlsum')
        do l=1,q
                j=1
                call AMPCal3(depart,xvec(nbobs),yvec(nbobs))
                call kho200(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=2
                call AMPCal3(depart,xvec(nbobs),yvec(nbobs))
                call kho100(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=3
                call AMPCal3(depart,xvec(nbobs),yvec(nbobs))
                call kho25(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=4
                call AMPCal3(depart,xvec(nbobs),yvec(nbobs))
                call CPG25(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=5
                call AMPCal3(depart,xvec(nbobs),yvec(nbobs))
                call joe200(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=6
                call AMPCal3(depart,xvec(nbobs),yvec(nbobs))
                call joe50(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=7
                call AMPCal3(depart,xvec(nbobs),yvec(nbobs))
                call joe25(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=8
                call AMPCal4(depart,xvec(nbobs),yvec(nbobs))
                call kho200(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=9
                call AMPCal4(depart,xvec(nbobs),yvec(nbobs))
                call kho100(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=10
                call AMPCal4(depart,xvec(nbobs),yvec(nbobs))
                call kho25(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=11
                call AMPCal4(depart,xvec(nbobs),yvec(nbobs))
                call CPG25(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=12
                call AMPCal4(depart,xvec(nbobs),yvec(nbobs))
                call joe100(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=13
                call AMPCal4(depart,xvec(nbobs),yvec(nbobs))
                call joe50(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=14
                call AMPCal4(depart,xvec(nbobs),yvec(nbobs))
                call joe25(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=15
                call AMFal3(depart,xvec(nbobs),yvec(nbobs))
                call kho200(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=16
                call AMFal3(depart,xvec(nbobs),yvec(nbobs))

                call kho100(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=17
                call AMFal3(depart,xvec(nbobs),yvec(nbobs))
                call kho25(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=18
                call AMFal3(depart,xvec(nbobs),yvec(nbobs))
                call CPG25(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=19
                call AMFal3(depart,xvec(nbobs),yvec(nbobs))
                call joe200(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=20
                call AMFal3(depart,xvec(nbobs),yvec(nbobs))
                call joe50(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=21
                call AMFal3(depart,xvec(nbobs),yvec(nbobs))
                call joe25(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=22
                call AMFal4(depart,xvec(nbobs),yvec(nbobs))
                call kho200(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=23
                call AMFal4(depart,xvec(nbobs),yvec(nbobs))
                call kho100(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
                Astarn1(ind,l)=Astarn(ind)
        enddo
        enddo
        call errl12(Astarn1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=24
                call AMFal4(depart,xvec(nbobs),yvec(nbobs))
                call kho25(xvec,yvec,tau(j,l),Astarn)
        do ind=1,d-1
```

```
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=25
        call AMFal4(depart,xvec(nbobs),yvec(nbobs))
        call CFG25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=26
        call AMFal4(depart,xvec(nbobs),yvec(nbobs))
        call joe200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=27
        call AMFal4(depart,xvec(nbobs),yvec(nbobs))
        call joe50(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=28
        call AMFal4(depart,xvec(nbobs),yvec(nbobs))
        call joe25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=29
        call A3PCal3(depart,xvec(nbobs),yvec(nbobs))
        call kho200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=30
        call A3PCal3(depart,xvec(nbobs),yvec(nbobs))
        call kho100(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=31
        call A3PCal3(depart,xvec(nbobs),yvec(nbobs))
        call kho25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=32
        call A3PCal3(depart,xvec(nbobs),yvec(nbobs))
        call CFG25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
endd
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=33
        call A3PCal3(depart,xvec(nbobs),yvec(nbobs))
        call joe200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=34
        call A3PCal3(depart,xvec(nbobs),yvec(nbobs))
        call joe50(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=35
        call A3PCal3(depart,xvec(nbobs),yvec(nbobs))
        call joe25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=36
        call A3PCal4(depart,xvec(nbobs),yvec(nbobs))
        call kho200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=37
        call A3PCal4(depart,xvec(nbobs),yvec(nbobs))

        call kho100(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=38
        call A3PCal4(depart,xvec(nbobs),yvec(nbobe))
        call kho25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=39
        call A3PCal4(depart,xvec(nbobe),yvec(nbobs))
        call CFG25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=40
        call A3PCal4(depart,xvec(nbobs),yvec(nbobs))
        call joe200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=41
        call A3PCal4(depart,xvec(nbobs),yvec(nbobs))
        call joe50(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=42
        call A3PCal4(depart,xvec(nbobs),yvec(nbobs))
        call joe25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=43
        call A3Fal3(depart,xvec(nbobs),yvec(nbobs))
        call kho200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=44
        call A3Fal3(depart,xvec(nbobs),yvec(nbobs))
        call kho100(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=45
        call A3Fal3(depart,xvec(nbobs),yvec(nbobs))
        call kho25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=46
        call A3Fal3(depart,xvec(nbobs),yvec(nbobs))
        call CFG25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=47
        call A3Fal3(depart,xvec(nbobs),yvec(nbobs))
        call joe200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=48
        call A3Fal3(depart,xvec(nbobs),yvec(nbobs))
        call joe50(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=49
        call A3Fal3(depart,xvec(nbobs),yvec(nbobs))
        call joe25(xvec,yvec,tau(j,l),Astarn)
```

```
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=50
        call AlFal4(depart,xvec(nbobs),yvec(nbobs))
        call kho200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=51
        call AlFal4(depart,xvec(nbobs),yvec(nbobs))
        call kho100(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=52
        call AlFal4(depart,xvec(nbobs),yvec(nbobs))
        call kho25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=53
        call AlFal4(depart,xvec(nbobs),yvec(nbobs))
        call CFG25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=54
        call AlFal4(depart,xvec(nbobs),yvec(nbobs))
        call joe200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=55
        call AlFal4(depart,xvec(nbobs),yvec(nbobs))
        call joe50(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=56
        call AlFal4(depart,xvec(nbobs),yvec(nbobs))
        call joe25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl3(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do j=1,56
        errL1bar(j)=0.d0
        errL1car(j)=0.d0
        taubar(j)=0.d0
        EQM(j)=0.d0
enddo
do j=1,56
        do l=1,q
                errL1bar(j)=errL1bar(j)+(erreurL11(j,l))/q
                errL1car(j)=errL1car(j)+(erreurL11(j,l)**2)
                taubar(j)=taubar(j)+(tau(j,l))/q
                EQM(j)=EQM(j)+((tau(j,l)-taustar)**2)/q
        enddo
enddo
        do j=1,56
        write(2,1003) taubar(j),EQM(j),errL1bar(j),errL1car(j)
1003    format(4(1X,F17.12),/)
        enddo
        do j=1,56
                do l=1,q
                        write(3,1006) tau(j,l),erreurL11(j,l)
1006                    format(1X,F17.12,1X,F17.12)


                enddo
        enddo
close(1)
close(2)
end
```

```
************************************************************************
PROGRAMME PRINCIPAL CALCULANT LES VALEURS DU TAU DE KENDALL ET DE L'ERREUR L1
 POUR DES ESTIMATIONS DE LA FONCTION A PAR LES METHODES DE JOE ET AL., DE
 KHOUDRAJI, ET DE CAPERAA ET AL., A PARTIR D'ECHANTILLONS DE TAILLE 2500 DE
 COPULE ARCHIMAX, OU LE TAU DE KENDALL THEORIQUE DE LA FONCTION A EST 0.257
************************************************************************
        program analyseA2
        include 'commun'
        integer q
        real*8 taustar
        parameter (q=100,taustar=0.257d0)
        integer nbobs,j,ind,l
        real*8 depart, xvec(n),yvec(n),taun,tau(d,d),erreurL1(d)
        real*8 Astarn(d),Astarn1(d,d),erreurL11(d,d)
        real*8 EQM(d),errL1bar(d),errL1car(d),taubar(d)
        depart=drand(1212)
        open(3,file='suma2')
        open(2,file='resA2sum')
        do l=1,q
                j=1
                call AMPCal1(depart,xvec(nbobs),yvec(nbobs))
                call kho200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=2
        call AMPCal1(depart,xvec(nbobs),yvec(nbobs))
        call kho100(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=3
        call AMPCal1(depart,xvec(nbobs),yvec(nbobs))
        call kho25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=4
        call AMPCal1(depart,xvec(nbobs),yvec(nbobs))
        call CFG25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=5
        call AMPCal1(depart,xvec(nbobs),yvec(nbobs))
        call joe200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=6
        call AMPCal1(depart,xvec(nbobs),yvec(nbobs))
        call joe50(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
enddo
do l=1,q
        j=7
        call AMPCal1(depart,xvec(nbobs),yvec(nbobs))
        call joe25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=8
        call AMPCal2(depart,xvec(nbobs),yvec(nbobs))
        call kho200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=9
        call AMPCal2(depart,xvec(nbobs),yvec(nbobs))
        call kho100(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=10
        call AMPCal2(depart,xvec(nbobs),yvec(nbobs))
        call kho25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=11
        call AMPCal2(depart,xvec(nbobs),yvec(nbobs))
        call CFG25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=12
        call AMPCal2(depart,xvec(nbobs),yvec(nbobs))
        call joe200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=13
        call AMPCal2(depart,xvec(nbobs),yvec(nbobs))
        call joe50(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call errl1(Astarn1,erreurL1)
```

```
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=14
        call AMPCal2(depart,xvec(nbobs),yvec(nbobs))
        call joe25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=15
        call AMFal1(depart,xvec(nbobs),yvec(nbobs))
        call kho200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=16
        call AMFal1(depart,xvec(nbobs),yvec(nbobs))
        call kho100(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=17
        call AMFal1(depart,xvec(nbobs),yvec(nbobs))
        call kho25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=18
        call AMFal1(depart,xvec(nbobs),yvec(nbobs))
        call CFG25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=19
        call AMFal1(depart,xvec(nbobs),yvec(nbobs))
        call joe200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=20
        call AMFal1(depart,xvec(nbobs),yvec(nbobs))
        call joe50(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=21
        call AMFal1(depart,xvec(nbobs),yvec(nbobs))
        call joe25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=22
        call AMFal2(depart,xvec(nbobs),yvec(nbobs))
        call kho200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=23
        call AMFal2(depart,xvec(nbobs),yvec(nbobs))
        call kho100(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=24
        call AMFal2(depart,xvec(nbobs),yvec(nbobs))
        call kho25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=25
        call AMFal2(depart,xvec(nbobs),yvec(nbobs))
        call CFG25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=26
        call AMFal2(depart,xvec(nbobs),yvec(nbobs))
        call joe200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=27
        call AMFal2(depart,xvec(nbobs),yvec(nbobs))
        call joe50(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=28
        call AMFal2(depart,xvec(nbobs),yvec(nbobs))
        call joe25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err11(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=29
        call A2PCal1(depart,xvec(nbobs),yvec(nbobs))
        call kho200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=30
        call A2PCal1(depart,xvec(nbobs),yvec(nbobs))
        call kho100(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo

do l=1,q
        j=31
        call A2PCal1(depart,xvec(nbobs),yvec(nbobs))
        call kho25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=32
        call A2PCal1(depart,xvec(nbobs),yvec(nbobs))
        call CFG25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=33
        call A2PCal1(depart,xvec(nbobs),yvec(nbobs))
        call joe200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=34
        call A2PCal1(depart,xvec(nbobs),yvec(nbobs))
        call joe50(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=35
        call A2PCal1(depart,xvec(nbobs),yvec(nbobs))
        call joe25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=36
        call A2PCal2(depart,xvec(nbobs),yvec(nbobs))
        call kho200(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=37
        call A2PCal2(depart,xvec(nbobs),yvec(nbobs))
        call kho100(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
call err12(Astarn1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=38
        call A2PCal2(depart,xvec(nbobs),yvec(nbobs))
        call kho25(xvec,yvec,tau(j,l),Astarn)
do ind=1,d-1
        Astarn1(ind,l)=Astarn(ind)
enddo
enddo
```

```
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=39
            call A2FCal2(depart,xvec(nbobs),yvec(nbobs))
            call CFG25(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=40
            call A2FCal2(depart,xvec(nbobs),yvec(nbobs))
            call joe200(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=41
            call A2FCal2(depart,xvec(nbobs),yvec(nbobs))
            call joe50(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=42
            call A2FCal2(depart,xvec(nbobs),yvec(nbobs))
            call joe25(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=43
            call A2Fal1(depart,xvec(nbobs),yvec(nbobs))
            call kho200(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=44
            call A2Fal1(depart,xvec(nbobs),yvec(nbobs))
            call kho100(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=45
            call A2Fal1(depart,xvec(nbobs),yvec(nbobs))
            call kho25(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=46
            call A2Fal1(depart,xvec(nbobs),yvec(nbobs))
            call CFG25(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=47
            call A2Fal1(depart,xvec(nbobs),yvec(nbobs))
            call joe200(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=48
            call A2Fal1(depart,xvec(nbobs),yvec(nbobs))
            call joe50(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=49
            call A2Fal1(depart,xvec(nbobs),yvec(nbobs))
            call joe25(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=50
            call A2Fal2(depart,xvec(nbobs),yvec(nbobs))
            call kho200(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=51
            call A2Fal2(depart,xvec(nbobs),yvec(nbobs))
            call kho100(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=52
            call A2Fal2(depart,xvec(nbobs),yvec(nbobs))
            call kho25(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=53
            call A2Fal2(depart,xvec(nbobs),yvec(nbobs))
            call CFG25(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=54
            call A2Fal2(depart,xvec(nbobs),yvec(nbobs))
            call joe200(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=55
            call A2Fal2(depart,xvec(nbobs),yvec(nbobs))
            call joe50(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do l=1,q
            j=56
            call A2Fal2(depart,xvec(nbobs),yvec(nbobs))
            call joe25(xvec,yvec,tau(j,l),Astarn)
      do ind=1,d-1
            Astarn1(ind,l)=Astarn(ind)
      enddo
      enddo
      call errl2(Astarn1,erreurL1)
      do l=1,q
            erreurL11(j,l)=erreurL1(l)
      enddo
      do j=1,56
            errL1bar(j)=0.d0
            errL1car(j)=0.d0
            taubar(j)=0.d0
            EQM(j)=0.d0
      enddo
      do j=1,56
            do l=1,q
                  errL1bar(j)=errL1bar(j)+(erreurL11(j,l))/q
                  errL1car(j)=errL1car(j)+(erreurL11(j,l)**2)
                  taubar(j)=taubar(j)+(tau(j,l))/q
                  EQM(j)=EQM(j)+((tau(j,l)-taustar)**2)/q
            enddo
      enddo
            do j=1,56
            write(2,1003) taubar(j),EQM(j),errL1bar(j),errL1car(j)
 1003       format(4(1X,F17.12),/)
            enddo
            do j=1,56
                  do l=1,q
                        write(3,1006) tau(j,l),erreurL11(j,l)
 1006                   format(1X,F17.12,1X,F17.12)
                  enddo
            enddo
      close(3)
      close(2)
      end
******************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec,yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE MIXTE (A1) AVEC theta=0.3 ET OU PHI EST LE
 GENERATEUR DE CLAYTON AVEC alpha=4/7
******************************************************************************
      subroutine AMPCal3(depart,xvec,yvec)
      include 'commun'
      integer nbobs,it,ip1,jt,nml
      real*8 xvec(n),yvec(n)
      real*8 depart,borne,z,A,nump,denomp,p,u,w,supCJ,auxil,x,y
      real*8 hmixed,simuAR,kphiCJ,phiCJ,phiinvCJ
      real*8 somme,sgnx,sgny,taun
      external  hmixed,simuAR,kphiCJ,phiCJ,phiinvCJ
      nbobs=0
1000  if (nbobs.lt.n) then
      parA=0.3d0
      alpha=0.57142857d0
      borne=1.d0+2.d0*parA/(4.d0-parA)
      z=simuAR(hmixed,borne)
      call depmixed(z,parA,A)
      nump=-2*parA*z*(1.d0-z)*(parA*z**2-parA*z+1.d0)
      denomp=parA**2*(z**4-2*z**3+z**2)
     *        + 4*parA*(z**2-z)+parA-1.d0
      p=nump/denomp
      u=drand(0)
      if (u.le.p) then
          w=drand(0)
          else
                c=1.d0
                supCJ=1.d0+1.d0/(c*alpha)
                w=simuAR(kphiCJ,supCJ)
       endif
          auxil=phiCJ(w)/A
          x=phiinvCJ(z*auxil)
          y=phiinvCJ((1.d0-z)*auxil)
      nbobs=nbobs+1
      xvec(nbobs)=x
      yvec(nbobs)=y
      goto 1000
      endif
       end
******************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec,yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE MIXTE (A1) AVEC theta=0.3 ET OU PHI EST LE
 GENERATEUR DE CLAYTON AVEC alpha=1.6
******************************************************************************
       subroutine AMPCal4(depart,xvec,yvec)
       include 'commun'
       integer nbobs,it,ip1,jt,nml
       real*8 xvec(n),yvec(n)
       real*8 depart,borne,z,A,nump,denomp,p,u,w,supCJ,auxil,x,y
       real*8 hmixed,simuAR,kphiCJ,phiCJ,phiinvCJ
       real*8 somme,sgnx,sgny,taun
```

```
        external  hmixed,simuAR,kphiCJ,phiCJ,phiinvCJ
        nbobs=0
1000    if (nbobs.lt.n) then
         parA=0.3d0
         alpha=1.6d0
         borne=1.d0+2.d0*parA/(4.d0-parA)
         z=simuAR(hmixed,borne)
        call depmixed(z,parA,A)
        nump=-2*parA*z*(1.d0-z)*(parA*z**2-parA*z+1.d0)
        denomp=parA**2*(z**4-2*z**3+z**2)
     *         + 4*parA*(z**2-z)+parA-1.d0
        p=nump/denomp
        u=drand(0)
        if (u.le.p) then
            w=drand(0)
            else
                 c=1.d0
                 supCJ=1.d0+1.d0/(c*alpha)
                 w=simuAR(kphiCJ,supCJ)
         endif
            auxil=phiCJ(w)/A
            x=phiinvCJ(z*auxil)
            y=phiinvCJ((1.d0-z)*auxil)
        nbobs=nbobs+1
        xvec(nbobs)=x
        yvec(nbobs)=y
        goto 1000
        endif
         end
*************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec,yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE MIXTE (A1) AVEC theta=0.3 ET OU PHI EST LE
 GENERATEUR DE FRANCK AVEC alpha=2.09
*************************************************************************
         subroutine AMFal3(depart,xvec,yvec)
         include 'commun'
         integer nbobs,it,ipl,jt,nml
         real*8 xvec(n),yvec(n)
         real*8 depart,borne,z,A,nump,denomp,p,u,w,auxil,x,y
         real*8 hmixed,simuAR,KphiFobs,phiF,phiinvF
         real*8 somme,sgnx,sgny,taun
         external  hmixed,simuAR,KphiFobs,phiF,phiinvF
         nbobs=0
1000    if (nbobs.lt.n) then
         parA=0.3d0
         alpha=2.09d0
         borne=1.d0+2.d0*parA/(4.d0-parA)
         z=simuAR(hmixed,borne)
        call depmixed(z,parA,A)
        nump=-2*parA*z*(1.d0-z)*(parA*z**2-parA*z+1.d0)
        denomp=parA**2*(z**4-2*z**3+z**2)
     *         + 4*parA*(z**2-z)+parA-1.d0
        p=nump/denomp
        u=drand(0)
        if (u.le.p) then
            w=drand(0)
            else
         c=1.d0
                 w=KphiFobs()
         endif
            auxil=phiF(w)/A
            x=phiinvF(z*auxil)
            y=phiinvF((1.d0-z)*auxil)
        nbobs=nbobs+1
        xvec(nbobs)=x
        yvec(nbobs)=y
        goto 1000
        endif
         end
*************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec,yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE MIXTE (A1) AVEC theta=0.3 ET OU PHI EST LE
 GENERATEUR DE FRANCK AVEC alpha=4.8
*************************************************************************
         subroutine AMFal4(depart,xvec,yvec)
         include 'commun'
         integer nbobs,it,ipl,jt,nml
         real*8 xvec(n),yvec(n)
         real*8 depart,borne,z,A,nump,denomp,p,u,w,auxil,x,y
         real*8 hmixed,simuAR,KphiFobs,phiF,phiinvF
         real*8 somme,sgnx,sgny,taun
         external  hmixed,simuAR,KphiFobs,phiF,phiinvF
         nbobs=0
1000    if (nbobs.lt.n) then
         parA=0.3d0
         alpha=4.8d0
         borne=1.d0+2.d0*parA/(4.d0-parA)
         z=simuAR(hmixed,borne)
        call depmixed(z,parA,A)
        nump=-2*parA*z*(1.d0-z)*(parA*z**2-parA*z+1.d0)
        denomp=parA**2*(z**4-2*z**3+z**2)
     *         + 4*parA*(z**2-z)+parA-1.d0
        p=nump/denomp
        u=drand(0)
        if (u.le.p) then
            w=drand(0)
            else
                 c=1.d0
                 w=KphiFobs()
         endif
            auxil=phiF(w)/A
            x=phiinvF(z*auxil)
            y=phiinvF((1.d0-z)*auxil)
        nbobs=nbobs+1
        xvec(nbobs)=x
        yvec(nbobs)=y
        goto 1000
        endif
         end
*************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec,yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE ASYMETRIQUE (A2) ET OU PHI EST LE
 GENERATEUR DE CLAYTON AVEC alpha=4/7
*************************************************************************
         subroutine A3FCal1(depart,xvec,yvec)
         include 'commun'
         integer nbobs,it,ipl,jt,nml
         real*8 xvec(n),yvec(n)
         real*8 depart,borne,z,A,nump,denomp,p,u,w,supCJ,auxil,x,y
         real*8 hmixed,simuAR,kphiCJ,phiCJ,phiinvCJ,fonc,fonc1,fonc2
         real*8 fonc3, fonc4
         real*8 KphiFobs,phiF,phiinvF,somme,sgnx,sgny,taun
         external hmixed,hA3,simuAR,kphiCJ,phiCJ,phiinvCJ
         external KphiFobs,phiF,phiinvF
         nbobs=0
1004    if (nbobs.lt.n) then
         borne=1.64865d0
         z=simuAR(hA3,borne)
        call depA3(z,A)
         if(z.le.0.2d0) then
         nump=z*(1.d0-z)*3.75d0
         fonc=1.875d0*z**2-0.75d0*z+1.d0
         fonc1=3.75d0*z-0.75d0
         fonc2=3.75d0
         fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
         fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
         denomp=(fonc3+fonc4)*fonc
         else
         nump=z*(1.d0-z)*0.234375d0
         fonc=0.0078125d0*(15.d0*z**2-6.d0*z+119.d0)
         fonc1=0.0078125d0*(30.d0*z-6.d0)
         fonc2=0.0078125d0*30.d0
         fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
         fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
         denomp=(fonc3+fonc4)*fonc
         endif
        p=nump/denomp
        u=drand(0)
        if (u.le.p) then
            w=drand(0)
            else
                 alpha=0.57142857d0
                 c=1.d0
                 supCJ=1.d0+1.d0/(c*alpha)
                 w=simuAR(kphiCJ,supCJ)
         endif
            auxil=phiCJ(w)/A
            x=phiinvCJ(z*auxil)
            y=phiinvCJ((1.d0-z)*auxil)
        nbobs=nbobs+1
        xvec(nbobs)=x
        yvec(nbobs)=y
        goto 1004
        endif
         end
*************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec,yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE ASYMETRIQUE (A2) ET OU PHI EST LE
 GENERATEUR DE CLAYTON AVEC alpha=1.6
*************************************************************************
         subroutine A3FCal4(depart,xvec,yvec)
         include 'commun'
         integer nbobs,it,ipl,jt,nml
         real*8 xvec(n),yvec(n)
         real*8 depart,borne,z,A,nump,denomp,p,u,w,supCJ,auxil,x,y
         real*8 hmixed,simuAR,kphiCJ,phiCJ,phiinvCJ,fonc,fonc1,fonc2
         real*8 fonc3, fonc4
         real*8 KphiFobs,phiF,phiinvF,somme,sgnx,sgny,taun
         external hmixed,hA3,simuAR,kphiCJ,phiCJ,phiinvCJ
         external KphiFobs,phiF,phiinvF
         nbobs=0
1004    if (nbobs.lt.n) then
         borne=1.64865d0
         z=simuAR(hA3,borne)
        call depA3(z,A)
         if(z.le.0.2d0) then
         nump=z*(1.d0-z)*3.75d0
         fonc=1.875d0*z**2-0.75d0*z+1.d0
         fonc1=3.75d0*z-0.75d0
         fonc2=3.75d0
         fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
         fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
         denomp=(fonc3+fonc4)*fonc
         else
         nump=z*(1.d0-z)*0.234375d0
         fonc=0.0078125d0*(15.d0*z**2-6.d0*z+119.d0)
         fonc1=0.0078125d0*(30.d0*z-6.d0)
         fonc2=0.0078125d0*30.d0
         fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
         fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
         denomp=(fonc3+fonc4)*fonc
         endif
        p=nump/denomp
        u=drand(0)
        if (u.le.p) then

                 alpha=1.6d0
                 c=1.d0
                 supCJ=1.d0+1.d0/(c*alpha)
                 w=simuAR(kphiCJ,supCJ)
         endif
            auxil=phiCJ(w)/A
            x=phiinvCJ(z*auxil)
            y=phiinvCJ((1.d0-z)*auxil)
        nbobs=nbobs+1
        xvec(nbobs)=x
        yvec(nbobs)=y
        goto 1004
        endif
         end
*************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec,yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE ASYMETRIQUE (A2) ET OU PHI EST LE
 GENERATEUR DE FRANCK AVEC alpha=2.09
*************************************************************************
         subroutine A3Fal3(depart,xvec,yvec)
         include 'commun'
         integer nbobs,it,ipl,jt,nml
         real*8 xvec(n),yvec(n)
         real*8 depart,borne,z,A,nump,denomp,p,u,w,auxil,x,y
         real*8 hmixed,simuAR,kphiCJ,phiCJ,phiinvCJ,fonc,fonc1,fonc2
         real*8 fonc3, fonc4
         real*8 KphiFobs,phiF,phiinvF,somme,sgnx,sgny,taun
         external hmixed,hA3,simuAR,kphiCJ,phiCJ,phiinvCJ
         external KphiFobs,phiF,phiinvF
         nbobs=0
1004    if (nbobs.lt.n) then
         borne=1.64865d0
         z=simuAR(hA3,borne)
        call depA3(z,A)
         if(z.le.0.2d0) then
         nump=z*(1.d0-z)*3.75d0
         fonc=1.875d0*z**2-0.75d0*z+1.d0
         fonc1=3.75d0*z-0.75d0
         fonc2=3.75d0
         fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
         fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
         denomp=(fonc3+fonc4)*fonc
         else
         nump=z*(1.d0-z)*0.234375d0
         fonc=0.0078125d0*(15.d0*z**2-6.d0*z+119.d0)
         fonc1=0.0078125d0*(30.d0*z-6.d0)
         fonc2=0.0078125d0*30.d0
         fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
         fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
         denomp=(fonc3+fonc4)*fonc
         endif
        p=nump/denomp
        u=drand(0)
        if (u.le.p) then
            w=drand(0)
            else
                 alpha=2.09d0
                 c=1.d0
                 w=KphiFobs()
         endif
            auxil=phiF(w)/A
            x=phiinvF(z*auxil)
            y=phiinvF((1.d0-z)*auxil)
        nbobs=nbobs+1
        xvec(nbobs)=x
        yvec(nbobs)=y
        goto 1004
        endif
         end
*************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec,yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE ASYMETRIQUE (A2) ET OU PHI EST LE
 GENERATEUR DE FRANCK AVEC alpha=4.8
*************************************************************************
         subroutine A3Fal4(depart,xvec,yvec)
         include 'commun'
         integer nbobs,it,ipl,jt,nml
         real*8 xvec(n),yvec(n)
         real*8 depart,borne,z,A,nump,denomp,p,u,w,auxil,x,y
         real*8 hmixed,simuAR,kphiCJ,phiCJ,phiinvCJ,fonc,fonc1,fonc2
         real*8 fonc3, fonc4
         real*8 KphiFobs,phiF,phiinvF,somme,sgnx,sgny,taun
         external hmixed,hA3,simuAR,kphiCJ,phiCJ,phiinvCJ
         external KphiFobs,phiF,phiinvF
         nbobs=0
1004    if (nbobs.lt.n) then
         borne=1.64865d0
         z=simuAR(hA3,borne)
        call depA3(z,A)
         if(z.le.0.2d0) then
         nump=z*(1.d0-z)*3.75d0
         fonc=1.875d0*z**2-0.75d0*z+1.d0
         fonc1=3.75d0*z-0.75d0
         fonc2=3.75d0
```

```
      fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
      fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
      denomp=(fonc3+fonc4)*fonc
      else
      nump=z*(1.d0-z)*0.234375d0
      fonc=0.0078125d0*(15.d0*z**2-6.d0*z+119.d0)
      fonc1=0.0078125d0*(30.d0*z-6.d0)
      fonc2=0.0078125d0*30.d0
      fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
      fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
      denomp=(fonc3+fonc4)*fonc
      endif
     p=nump/denomp
     u=drand(0)
     if (u.le.p) then
         w=drand(0)
         else
                alpha=4.8d0
                c=1.d0
                w=KphiFobs()
      endif
         auxil=phiF(w)/A
         x=phiinvF(z*auxil)
         y=phiinvF((1.d0-z)*auxil)
     nbobs=nbobs+1
     xvec(nbobs)=x
     yvec(nbobs)=y
     goto 1004
     endif
      end
****************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec.yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE MIXTE (A3) AVEC theta=0.668 ET OU PHI EST LE
 GENERATEUR DE CLAYTON AVEC alpha=0.972
****************************************************************************
      subroutine AMPCal1(depart.xvec,yvec)
      include 'commun'
      integer nbobs,it,ip1,jt,nml
      real*8 xvec(n),yvec(n)
      real*8 depart.borne.z.A.nump.denomp.p.u.w.supCJ.auxil.x.y
      real*8 hmixed.simuAR.kphiCJ.phiCJ.phiinvCJ
      real*8 somme.sgnx.sgny.taun
      external  hmixed,simuAR.kphiCJ.phiCJ.phiinvCJ
      nbobs=0
1000  if (nbobs.lt.n) then
      parA=0.668d0
      alpha=0.972d0
      borne=1.d0+2.d0*parA/(4.d0-parA)
      z=simuAR(hmixed,borne)
     call depmixed(z,parA,A)
     nump=-2*parA*z*(1.d0-z)*(parA*z**2-parA*z+1.d0)
     denomp=parA**2*(z**4-2*z**3+z**2)
   +        + 4*parA*(z**2-z)+parA-1.d0
     p=nump/denomp
     u=drand(0)
     if (u.le.p) then
         w=drand(0)
         else
                c=1.d0
                supCJ=1.d0+1.d0/(c*alpha)
                w=simuAR(kphiCJ,supCJ)
      endif
         auxil=phiCJ(w)/A
         x=phiinvCJ(z*auxil)
         y=phiinvCJ((1.d0-z)*auxil)
     nbobs=nbobs+1
     xvec(nbobs)=x
     yvec(nbobs)=y
     goto 1000
     endif
      end
****************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec.yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE MIXTE (A3) AVEC theta=0.668 ET OU PHI EST LE
 GENERATEUR DE CLAYTON AVEC alpha=2.953
****************************************************************************
      subroutine AMPCal2(depart,xvec,yvec)
      include 'commun'
      integer nbobs.it.ip1.jt,nml
      real*8 xvec(n),yvec(n)
      real*8 depart.borne.z.A.nump.denomp.p.u.w.supCJ.auxil.x,y
      real*8 hmixed.simuAR.kphiCJ.phiCJ.phiinvCJ
      real*8 somme.sgnx.sgny.taun
      external  hmixed.simuAR.kphiCJ.phiCJ.phiinvCJ
      nbobs=0
1000  if (nbobs.lt.n) then
      parA=0.668d0
      alpha=2.953d0
      borne=1.d0+2.d0*parA/(4.d0-parA)
      z=simuAR(hmixed,borne)
     call depmixed(z.parA.A)
     nump=-2*parA*z*(1.d0-z)*(parA*z**2-parA*z+1.d0)
     denomp=parA**2*(z**4-2*z**3+z**2)
   +        + 4*parA*(z**2-z)+parA-1.d0
     p=nump/denomp
     u=drand(0)
     if (u.le.p) then
         w=drand(0)
         else
                c=1.d0
                supCJ=1.d0+1.d0/(c*alpha)
                w=simuAR(kphiCJ.supCJ)
      endif
         auxil=phiCJ(w)/A
         x=phiinvCJ(z*auxil)
         y=phiinvCJ((1.d0-z)*auxil)
     nbobs=nbobs+1
     xvec(nbobs)=x
     yvec(nbobs)=y
     goto 1000
     endif
      end
****************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec,yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE MIXTE (A3) AVEC theta=0.668 ET OU PHI EST LE
 GENERATEUR DE FRANCK AVEC alpha=3.23
****************************************************************************
      subroutine AMFal1(depart.xvec.yvec)
      include 'commun'
      integer nbobs,it,ip1,jt,nml
      real*8 xvec(n),yvec(n)
      real*8 depart.borne.z,A,nump,denomp,p,u,w,auxil,x,y
      real*8 hmixed.simuAR.KphiFobs.phiF.phiinvF
      real*8 somme.sgnx.sgny.taun
      external  hmixed.simuAR.KphiFobs.phiF.phiinvF
      nbobs=0
1000  if (nbobs.lt.n) then
      parA=0.668d0
      alpha=3.23d0
      borne=1.d0+2.d0*parA/(4.d0-parA)
      z=simuAR(hmixed.borne)
     call depmixed(z,parA,A)
     nump=-2*parA*z*(1.d0-z)*(parA*z**2-parA*z+1.d0)
     denomp=parA**2*(z**4-2*z**3+z**2)
   +        + 4*parA*(z**2-z)+parA-1.d0
     p=nump/denomp
     u=drand(0)
     if (u.le.p) then
         w=drand(0)
         else
                c=1.d0
                w=KphiFobs()
      endif
         auxil=phiF(w)/A
         x=phiinvF(z*auxil)
         y=phiinvF((1.d0-z)*auxil)
    nbobs=nbobs+1
     xvec(nbobs)=x
     yvec(nbobs)=y
     goto 1000
     endif
      end
****************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec,yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE MIXTE (A3) AVEC theta=0.668 ET OU PHI EST LE
 GENERATEUR DE FRANCK AVEC alpha=7.83
****************************************************************************
      subroutine AMFal2(depart,xvec,yvec)
      include 'commun'
      integer nbobs.it.ip1.jt,nml
      real*8 xvec(n),yvec(n)
      real*8 depart,borne.z.A,nump,denomp.p.u.w.auxil.x,y
      real*8 hmixed.simuAR.kphiCJ,phiCJ,phiinvCJ.KphiFobs.phiF.phiinvF
      real*8 somme.sgnx.sgny.taun
      external  hmixed.simuAR.kphiCJ.phiCJ.phiinvCJ
      external KphiFobs.phiF.phiinvF
      nbobs=0
1000  if (nbobs.lt.n) then
      parA=0.668d0
      alpha=7.83d0
      borne=1.d0+2.d0*parA/(4.d0-parA)
      z=simuAR(hmixed.borne)
     call depmixed(z.parA.A)
     nump=-2*parA*z*(1.d0-z)*(parA*z**2-parA*z+1.d0)
     denomp=parA**2*(z**4-2*z**3+z**2)
   +        + 4*parA*(z**2-z)+parA-1.d0
     p=nump/denomp
     u=drand(0)
     if (u.le.p) then
         w=drand(0)
         else
                c=1.d0
                w=KphiFobs()
      endif
         auxil=phiF(w)/A
         x=phiinvF(z*auxil)
         y=phiinvF((1.d0-z)*auxil)
     nbobs=nbobs+1
     xvec(nbobs)=x
     yvec(nbobs)=y
     goto 1000
     endif
      end
****************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec.yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE ASYMETRIQUE (A4) ET OU PHI EST LE GENERATEUR
 DE CLAYTON AVEC alpha=0.972
****************************************************************************
      subroutine A2PCal1(depart,xvec,yvec)
      include 'commun'
      integer nbobs,it,ip1,jt,nml
      real*8 xvec(n),yvec(n)
      real*8 depart,borne,z,A,nump,denomp.p.u.w.supCJ.auxil,x,y
      real*8 hmixed,simuAR,kphiCJ,phiCJ,phiinvCJ,fonc,fonc1.fonc2
      real*8 fonc3, fonc4
      real*8 KphiFobs,phiF,phiinvF,somme,sgnx,sgny,taun
      external hmixed,hA2,simuAR,kphiCJ,phiCJ,phiinvCJ
      external KphiFobs,phiF,phiinvF
      nbobs=0
1004  if (nbobs.lt.n) then
      borne=1.8
      z=simuAR(hA2,borne)
     call depA2(z,A)
      if(z.lt.0.33333d0) then
      nump=z*(1.d0-z)*3.d0
      fonc=1.5d0*z**2-z+1.d0
      fonc1=3.d0*z-1.d0
      fonc2=3.d0
      fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
      fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
      denomp=(fonc3+fonc4)*fonc
      else
      nump=z*(1.d0-z)*0.75d0
      fonc=0.125d0*(3.d0*z**2-2.d0*z+7.d0)
      fonc1=0.125d0*(6.d0*z-2.d0)
      fonc2=0.125d0*6.d0
      fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
      fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
      denomp=(fonc3+fonc4)*fonc
      endif
     p=nump/denomp
     u=drand(0)
     if (u.le.p) then
         w=drand(0)
         else
                alpha=0.972
                c=1.d0
                supCJ=1.d0+1.d0/(c*alpha)
                w=simuAR(kphiCJ,supCJ)
      endif
         auxil=phiCJ(w)/A
         x=phiinvCJ(z*auxil)
         y=phiinvCJ((1.d0-z)*auxil)
     nbobs=nbobs+1
     xvec(nbobs)=x
     yvec(nbobs)=y
     goto 1004
     endif
      end
****************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec.yvec) DE COPULE ARCHIMAX. OU A EST
 LA FONCTION DE DEPENDANCE ASYMETRIQUE (A4) ET OU PHI EST LE GENERATEUR
 DE CLAYTON AVEC alpha=2.953
****************************************************************************
      subroutine A2PCal2(depart,xvec,yvec)
      include 'commun'
      integer nbobs.it.ip1.jt.nml
      real*8 xvec(n),yvec(n)
      real*8 depart.borne.z.A.nump.denomp.p.u.w.supCJ.auxil,x,y
      real*8 hmixed,simuAR,kphiCJ,phiCJ,phiinvCJ,fonc,fonc1,fonc2
      real*8 fonc3, fonc4
      real*8 KphiFobs.phiF.phiinvF.somme,sgnx,sgny,taun
      external hmixed,hA2,simuAR,kphiCJ,phiCJ,phiinvCJ
      external KphiFobs.phiF.phiinvF
      nbobs=0
1004  if (nbobs.lt.n) then
      borne=1.8
      z=simuAR(hA2,borne)
     call depA2(z,A)
      if(z.lt.0.33333d0) then
      nump=z*(1.d0-z)*3.d0
      fonc=1.5d0*z**2-z+1.d0
      fonc1=3.d0*z-1.d0
      fonc2=3.d0
      fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
      fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
      denomp=(fonc3+fonc4)*fonc
      else
      nump=z*(1.d0-z)*0.75d0
      fonc=0.125d0*(3.d0*z**2-2.d0*z+7.d0)
      fonc1=0.125d0*(6.d0*z-2.d0)
      fonc2=0.125d0*6.d0
      fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
      fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
      denomp=(fonc3+fonc4)*fonc
      endif
     p=nump/denomp
     u=drand(0)
     if (u.le.p) then
        w=drand(0)
         else
                alpha=2.953
                c=1.d0
                supCJ=1.d0+1.d0/(c*alpha)
                w=simuAR(kphiCJ.supCJ)
      endif
         auxil=phiCJ(w)/A
         x=phiinvCJ(z*auxil)
         y=phiinvCJ((1.d0-z)*auxil)
     nbobs=nbobs+1
```

```
      xvec(nbobs)=x
      yvec(nbobs)=y
      goto 1004
      endif
       end
***************************************************************************
PROCEDURE GENERANT 1500 COUPLES (xvec,yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE ASYMETRIQUE (A4) ET OU PHI EST LE GENERATEUR
 DE FRANCK AVEC alpha=3.23
***************************************************************************
       subroutine A2Fal1(depart,xvec,yvec)
       include 'common'
       integer nbobs,it,ip1,jt,nm1
       real*8 xvec(n),yvec(n)
       real*8 depart,borne,z,A,nump,denomp,p,u,w,auxil,x,y
       real*8 bmixed,simuAR,kphiCJ,phiCJ,phiinvCJ,fonc,fonc1,fonc2
       real*8 fonc3, fonc4
       real*8 KphiFobs,phiF,phiinvF,somme,sgnx,sgny,taun
       external bmixed,hA2,simuAR,kphiCJ,phiCJ,phiinvCJ
       external KphiFobs,phiF,phiinvF
       nbobs=0
1004  if (nbobs.lt.n) then
       borne=1.8
       z=simuAR(hA2,borne)
      call depA2(z,A)
       if(z.lt.0.33333d0) then
       nump=z*(1.d0-z)*3.d0
       fonc=1.5d0*z**2-z+1.d0
       fonc1=3.d0*z-1.d0
       fonc2=3.d0
       fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
       fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
       denomp=(fonc3+fonc4)*fonc
       else
       nump=z*(1.d0-z)*0.75d0
       fonc=0.125d0*(3.d0*z**2-2.d0*z+7.d0)
       fonc1=0.125d0*(6.d0*z-2.d0)
       fonc2=0.125d0*6.d0
       fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
       fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
       denomp=(fonc3+fonc4)*fonc
       endif
      p=nump/denomp
      u=drand(0)
      if (u.le.p) then
          w=drand(0)
          else
                alpha=3.23d0
                c=1.d0
                w=KphiFobs()
       endif
          auxil=phiF(w)/A
          x=phiinvF(z*auxil)
          y=phiinvF((1.d0-z)*auxil)
      nbobs=nbobs+1
      xvec(nbobs)=x
      yvec(nbobs)=y
      goto 1004
      endif
       end
***************************************************************************
PROCEDURE GENERANT 2500 COUPLES (xvec,yvec) DE COPULE ARCHIMAX, OU A EST
 LA FONCTION DE DEPENDANCE ASYMETRIQUE (A4) ET OU PHI EST LE GENERATEUR DE
 FRANCK AVEC alpha=7.83
***************************************************************************
       subroutine A2Fal2(depart,xvec,yvec)
       include 'common'
       integer nbobs,it,ip1,jt,nm1
       real*8 xvec(n),yvec(n)
       real*8 depart,borne,z,A,nump,denomp,p,u,w,auxil,x,y
       real*8 bmixed,simuAR,kphiCJ,phiCJ,phiinvCJ,fonc,fonc1,fonc2
       real*8 fonc3, fonc4
       real*8 KphiFobs,phiF,phiinvF,somme,sgnx,sgny,taun
       external bmixed,hA2,simuAR,kphiCJ,phiCJ,phiinvCJ
       external KphiFobs,phiF,phiinvF
       nbobs=0
1004  if (nbobs.lt.n) then
       borne=1.8
       z=simuAR(hA2,borne)
      call depA2(z,A)
       if(z.lt.0.33333d0) then
       nump=z*(1.d0-z)*3.d0
       fonc=1.5d0*z**2-z+1.d0
       fonc1=3.d0*z-1.d0
       fonc2=3.d0
       fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
       fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
       denomp=(fonc3+fonc4)*fonc
       else
       nump=z*(1.d0-z)*0.75d0
       fonc=0.125d0*(3.d0*z**2-2.d0*z+7.d0)
       fonc1=0.125d0*(6.d0*z-2.d0)
       fonc2=0.125d0*6.d0
       fonc3=1.d0+(1.d0-2.d0*z)*fonc1/fonc
       fonc4=z*(1-z)*(fonc2/fonc-(fonc1/fonc)**2)
       denomp=(fonc3+fonc4)*fonc
       endif
      p=nump/denomp
      u=drand(0)
      if (u.le.p) then
          w=drand(0)
          else
                alpha=7.83d0
                c=1.d0
                w=KphiFobs()
       endif
          auxil=phiF(w)/A
          x=phiinvF(z*auxil)
          y=phiinvF((1.d0-z)*auxil)
      nbobs=nbobs+1
      xvec(nbobs)=x
      yvec(nbobs)=y
      goto 1004
      endif
       end

***************************************************************************
PROCEDURE CALCULANT LE TAU DE KENDALL D'UNE DISTRIBUTION DE VE, DE FONCTION
 DE DEPENDANCE An CONNUE EN CHAQUE POINT D'UNE SUBDIVISION sub DE PAS pas
 DE [0,1], DE TAILLE d.
***************************************************************************

      subroutine tauE(sub,pas,av,tautrap,tausimpson,aux)
      include 'common'
      real*8 tautrap,tausimpson, pas
      real*8 av(d), sub(d), aux(d), d2av(d)
      integer l, j
      do 11 l=2,d-2
            d2av(l)=(av(l+1)+av(l-1)-2.d0*av(l))/pas**2
            aux(l)=sub(l)*(1-sub(l))*d2av(l)/av(l)
 11   continue
      aux(1)=0.d0
      aux(d-1)=0.d0
      aux(d)=0.d0
      tautrap=0.d0
      do 12 j=1,d-2
            tautrap=tautrap+pas*(aux(j)+aux(j+1))/2.d0
 12         continue
      j=1
      tausimpson=0.d0
101   if (j.lt.d-1) then
          tausimpson=tausimpson
     +           + pas*(aux(j)+4.d0*aux(j+1)+aux(j+2))/3.d0
          j=j+2
          goto 101
          endif

      close(1)
      end

***************************************************************************
PROCEDURE CALCULANT LE VECTEUR 'irank' DES RANGS D'UN VECTEUR DE TAILLE 'n' A
 PARTIR DU VECTEUR 'indx1' DES ANTI-RANGS, RESULTAT DE LA PROCEDURE 'indexx.f'
***************************************************************************
       subroutine rank(n,indx1,irank)
       integer n, indx1(n), irank(n)
       integer j
       do 10 j=1,n
             irank(indx1(j))=j
10     continue
       end
***************************************************************************
PROCEDURE QUI TRIE PAR ORDRE CROISSANT UN VECTEUR ARRIN DE TAILLE NN, ET STOCKE
 LES ANTIRANGS DANS LE VECTEUR INDX.
***************************************************************************
      SUBROUTINE INDEXX(NN,ARRIN,INDX)
      real*8 arrin(nn)
      integer nn, indx(nn),j,ir,indxt,l,i
      DIMENSION INDX(NN)
      real*8 q
      DO 11 J=1,NN
        INDX(J)=J
11    CONTINUE
      L=NN/2+1
      IR=NN
10    CONTINUE
        IF(L.GT.1)THEN
          L=L-1
          INDXT=INDX(L)
          Q=ARRIN(INDXT)
        ELSE
          INDXT=INDX(IR)
          Q=ARRIN(INDXT)
          INDX(IR)=INDX(1)
          IR=IR-1
          IF(IR.EQ.1)THEN
            INDX(1)=INDXT
            RETURN
          ENDIF
        ENDIF
        I=L
        J=L+L
20      IF(J.LE.IR)THEN
          IF(J.LT.IR)THEN
            IF(ARRIN(INDX(J)).LT.ARRIN(INDX(J+1)))J=J+1
          ENDIF
          IF(Q.LT.ARRIN(INDX(J)))THEN
            INDX(I)=INDX(J)
            I=J
            J=J+J
          ELSE
            J=IR+1
          ENDIF
        GO TO 20
        ENDIF
        INDX(I)=INDXT
      GO TO 10
      END
***************************************************************************
FONCTION RETOURNANT LA VALEUR DE LA FONCTION PHI en t, ou PHI = (PSI)^c, avec
 c >= 1, et ou PSI  = generateur de la copule archimedienne de Cook et Johnson
 de parametre alpha, alpha > 0.
***************************************************************************

       real*8 function phiCJ(t)
       real*8 t
       include 'commun'
       phiCJ=((t**(-alpha)-1.d0)/alpha)**c
       end
***************************************************************************
FONCTION RETOURNANT LA VALEUR DE LA FONCTION PHI en t, ou PHI = (PSI)^c, avec
c >= 1, et ou PSI  = generateur de la copule archimedienne de Frank de
parametre alpha \in R.
***************************************************************************
       real*8 function phiF(t)
       real*8 t
       include 'commun'
       phiF=(-log((1.d0-exp(-t*alpha))/(1-exp(-alpha))))**c
       end
***************************************************************************
FONCTION RETOURNANT LA VALEUR DE L'INVERSE DE LA FONCTION PHI en t ,ou PHI
 = (PSI)^c, avec c >= 1, et ou PSI  = generateur de la copule archimedienne de
 Cook et Johnson de parametre alpha, alpha > 0.
***************************************************************************
       real*8 function phiinvCJ(t)
       real*8 t
       include 'commun'
       phiinvCJ=(1.d0+alpha*t**(1.d0/c))**(-1.d0/alpha)
       end
***************************************************************************
FONCTION RETOURNANT LA VALEUR DE L'INVERSE DE LA FONCTION PHI en t, ou PHI
 = (PSI)^c, avec c >= 1, et ou PSI  = generateur de la copule archimedienne
 de Frank de parametre alpha \in R.
***************************************************************************

        real*8 function phiinvF(t)
        real*8 t
        include 'commun'
        phiinvF=(-1.d0/alpha)*log(1-exp(-t**(1.d0/c))*(1-exp(-alpha)))
        end
***************************************************************************
 FONCTION RETOURNANT LA VALEUR EN t DE LA DENSITE kphi, OU phi = (psi)^c, avec
 c >= 1, et psi  = GENERATEUR DE COOK & JOHNSON
 kphi(t)=1+(1-(alpha+1)t**alpha)/(c*alpha)
***************************************************************************

        real*8 function kphiCJ(t)
        real*8 t,auxl
        include 'commun'
        auxl=1.d0 + (1.d0 - (alpha+1.d0)*t**alpha)/(c*alpha)
        kphiCJ=auxl
        end
***************************************************************************
                                   FONCTION RETOURNANT UNE OBSERVATION ISSUE DE LA F
ONCTION DE REPARTITION Kphi,
 OU phi = (psi)^c, avec c >=1, et psi  = GENERATEUR DE FRANK on utilise le fait
 que si (X,Y)~ C, ou C est une copule archimedienne de generateur phi,alors
C(X,Y)~ Kphi. Pour generer (X,Y), on utilise l'algorithme de Genest & MacKay
(1986).
***************************************************************************
        real*8 function KphiFobs()
        real*8  x1, x2, t1, t2, aaux
        include 'commun'
        real*8 dphiF, inverse, zbrent
        external dphiF, inverse, zbrent
        x1=drand(0)
        x2=drand(0)
        ord=0.d0
        t1=dphiF(x1)
        aaux=t1/x2
        t2=inverse(aaux)
        KphiFobs=t2
        end
***************************************************************************
FONCTION CALCULANT L'INVERSE EN UN POINT D'UNE FONCTION dphiF DECROISSANTE SUR
 [a,b], avec une erreur de epsilon. (a,b, et epsilon sont a specifier dans la
 fonction).
***************************************************************************
        real*8 function inverse(u)
        integer nbiter
        real*8 epsilon, a, b, xg, xd, u, y, mil
        real*8 dphiF
        external dphiF
        epsilon=5.d-3
        a=0.d0
        b=1.d0
        nbiter=0
        mil=(a+b)/2
        y=dphiF(mil)
        xg=a
        xd=b
```

```fortran
10    if ((abs(y-u).ge.epsilon).and.(nbiter.le.1000)) then
         if (y.gt.u) then
                      xd=mil
                   else
                      xg=mil
                   endif
        mil=(xg+xd)/2
        y=dphiF(mil)
        nbiter=nbiter+1
        goto 10
       endif
       if (nbiter.ge.1000) then
          write(6,*)'trop d iterations ! '
          endif
      inverse=mil
      end
*****************************************************************************
      real*8 FUNCTION ZBRENT(FUNC,X1,X2,TOL)
      integer itmax,iter
      real*8 x1,x2,tol,eps,a, b,c,d,e,fa,fb,fc,xm,tol1,s,
     *       p,q,r
      real*8 func
      external func
      PARAMETER (ITMAX=100,EPS=3.D-8)
      A=X1
      B=X2
      FA=FUNC(A)
      FB=FUNC(B)
      IF(FB*FA.GT.0.d0) PAUSE 'Root must be bracketed for ZBRENT.'
      FC=FB
      DO 11 ITER=1,ITMAX
        IF(FB*FC.GT.0.d0) THEN
          C=A
          FC=FA
          D=B-A
          E=D
        ENDIF
        IF(ABS(FC).LT.ABS(FB)) THEN
          A=B
          B=C
          C=A
          FA=FB
          FB=FC
          FC=FA
        ENDIF
        TOL1=2.d0*EPS*ABS(B)+0.5d0*TOL
        XM=.5d0*(C-B)
        IF(ABS(XM).LE.TOL1 .OR. FB.EQ.0.d0)THEN
          ZBRENT=B
          RETURN
        ENDIF
        IF(ABS(E).GE.TOL1 .AND. ABS(FA).GT.ABS(FB)) THEN
          S=FB/FA
          IF(A.EQ.C) THEN
            P=2.d0*XM*S
            Q=1.d0-S
          ELSE
            Q=FA/FC
            R=FB/FC
            P=S*(2.d0*XM*Q*(Q-R)-(B-A)*(R-1.d0))
           Q=(Q-1.d0)*(R-1.d0)*(S-1.d0)
         ENDIF
         IF(P.GT.0.d0) Q=-Q
         P=ABS(P)
         IF(2.d0*P .LT. MIN(3.d0*XM*Q-ABS(TOL1*Q),ABS(E*Q))) THEN
           E=D
           D=P/Q
         ELSE
           D=XM
           E=D
         ENDIF
       ELSE
         D=XM
         E=D
       ENDIF
       A=B
       FA=FB
       IF(ABS(D) .GT. TOL1) THEN
         B=B+D
       ELSE
         B=B+SIGN(TOL1,XM)
       ENDIF
       FB=FUNC(B)
11    CONTINUE
      PAUSE 'ZBRENT exceeding maximum iterations.'
      ZBRENT=B
      RETURN
      END
*****************************************************************************
FONCTION CALCULANT (-ord + phi'(t)), ou phi' est LA DERIVEE  EN UN POINT t  DE
 LA FONCTION 'phi', OU phi = psi^c, avec  'psi' = GENERATEUR  DE FRANK, et c
 >=1.
*****************************************************************************

      real*8  function dphiF(t)
      real*8 t, s1, s2,auxili
      include 'commun'
      auxili=exp(-alpha)
      s1=exp(-t*alpha)
      s2=-(-log((1.d0-s1)/(1.d0-auxili)))**(c-1.d0) *c*alpha*s1/(1.d0-s1)
      dphiF=s2-ord
      end
*****************************************************************************
                  FONCTION RETOURNANT LA VALEUR EN t DE LA DENSITE h DE LA V.
A. 'Z', POUR UNE
 FONCTION DE DEPENDANCE DU MODELE MIXTE, A = A(th), 0 <= th <=1.
*****************************************************************************
      real*8 function hmixed(t)
      real*8 t,th,num,denom
      include 'commun'
      th=parA
      num=th**2*(-t**4+2*t**3-t**2)+4*th*(-t**2+t)+1-th
      denom=(th*t**2-th*t+1)**2
      hmixed=num/denom
      end
*****************************************************************************
PROCEDURE CALCULANT LA VALEUR EN t DE A, OU A EST LA FONCTION DE DEPENDANCE DU
 MODELE MIXTE DE TAWN (1988), DE PARAMETRE 0<=pA<=1
*****************************************************************************
      subroutine depmixed(t,pA,A1)
      real*8 t,pA,A1
      include 'commun'
      A1=pA*t**2 - pA*t +1.d0
      end
*****************************************************************************
PROCEDURE CALCULANT LA VALEUR EN t DE A, OU A EST LA FONCTION DE DEPENDANCE DU
 MODELE A4
*****************************************************************************
      subroutine depA2(t,A1)
      real*8 t,A1
      include 'commun'
      if (t.lt.0.333333d0) then
              A1=3.d0*(t**2)/2.d0-t+1.d0
       else
              A1=0.125d0*(3.d0*t**2.d0-2.d0*t+7.d0)
       endif
      end
*****************************************************************************
FONCTION RETOURNANT LA VALEUR EN t DE LA DENSITE h DE LA V. A. 'Z', POUR UNE
 FONCTION DE DEPENDANCE DU MODELE A4, A = A(th), 0 <= th <=1.
*****************************************************************************
      real*8 function hA2(t)
      real*8 t,A,A1,A2
      include 'commun'
      if (t.lt.0.33333d0) then
              A=1.5d0*t**2-t+1.d0
              A1=1.5d0*2.d0*t-1.d0
              A2=1.5d0*2.d0
      else
              A=(3.d0*t**2-2.d0*t+7.d0)/8.d0
              A1=(6.d0*t-2.d0)/8.d0
              A2=6/8
      endif
      hA2=1+(1-2*t)*A1/A+t*(1-t)*(A2/A-((A1/A)**2))
      end
*****************************************************************************
FONCTION RETOURNANT LA VALEUR EN t DE LA DENSITE h DE LA V. A. 'Z', POUR UNE
FONCTION DE DEPENDANCE DU MODELE A2, A = A(th), 0 <= th <=1.
*****************************************************************************

      real*8 function hA3(t)
      real*8 t,A,A1,A2
      include 'commun'
       if (t.lt.0.2d0) then
              A=1.875d0*t**2-0.75d0*t+1.d0
              A1=3.75d0*t-0.75d0
              A2=3.75d0
       else
              A=(15.d0*t**2-6.d0*t+119.d0)/128.d0
              A1=(30.d0*t-6.d0)/128.d0
              A2=0.234375d0
       endif
      hA3=1+(1-2*t)*A1/A+t*(1-t)*(A2/A-((A1/A)**2))
      end
*****************************************************************************
 PROCEDURE CALCULANT LA VALEUR EN t DE A, OU A EST LA FONCTION DE DEPENDANCE DU
MODELE A2
*****************************************************************************

      subroutine depA3(t,A1)
      real*8 t,A1
      include 'commun'
      if (t.le.0.2d0) then
              A1=1.875d0*t**2-0.75d0*t+1
       else
              A1=0.0078125d0*(15.d0*t**2.d0-6.d0*t+119.d0)
       endif
      end
*****************************************************************************
FONCTION RETOURNANT UNE OBSERVATION ISSUE D'UNE V. A. DE DENSITE f, BORNEE PAR
 borne, EN UTILISANT LA METHODE D'ACCEPTATION/REJET.
*****************************************************************************
      real*8 function simuAR(f, borne)
      integer test
      real*8 z1, borne, zstar, u1, aux
      include 'commun'
      real*8 f
      external f
      test=0
1001  if(test.eq.0) then
            zstar=drand(0)
            u1=drand(0)
            aux=f(zstar)
            if (aux.ge.borne*u1) then
                                  z1=zstar
                                  test=1
                               endif
            goto 1001
       endif
      simuAR=z1
      end
*****************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE JOE ET AL. AVEC UN SEUIL DE 200/2500
*****************************************************************************
       subroutine joe25(xvec,yvec,taut7,Astarn)
       include 'commun'
      integer rgx(n), rgy(n), antirgx(n), antirgy(n),antirgt(n),
     *       ind,nstar,i,j
       real*8 xvec(n),yvec(n), xord(n), yord(n),pas,tord(n)
       real*8 Astarn(d),t(d),B,e(n)
       real*8 sum

       real*8 taut7,taus,integ(d)
       real*8 Q,z1(n),z2(n)
       do 199 ind=1,d
             t(ind)=dble(ind)/dble(d)
199          continue
       pas=t(2)-t(1)
       nstar=25.d0
       call indexx(n,xvec,antirgx)
       call indexx(n,yvec,antirgy)
       do 301 i=1,n
             xord(i)=xvec(antirgx(i))
             yord(i)=yvec(antirgy(i))
301          continue
      call rank(n,antirgx,rgx)
      call rank(n,antirgy,rgy)
      do 302 i=1,n
             z1(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
             z2(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
302    continue
       do 305 ind=1,d-1
              B=(1-t(ind))/t(ind)
       do 306 i=1,n
       Q=B*z2(i)
              if (z1(i).GT.Q) then
                     e(i)=z1(i)/n
              else
                     e(i)=Q/n
              endif
306    enddo
       call indexx(n,e,antirgt)
       do 303 i=1,n
              tord(i)=e(antirgt(i))
303    continue
       sum=0.d0
       j=25.d0
       do 304 i=2476,2475+10
       sum=tord(i)*j+sum
       j=j-1.d0
304    enddo
       Astarn(ind)=t(ind)*sum/10.d0
305    enddo
       call tauK(t,pas,Astarn,taut7,taus,integ)
       end
*****************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE JOE ET AL. AVEC UN SEUIL DE 100/2500
*****************************************************************************
       subroutine joe50(xvec,yvec,taut7,Astarn)
       include 'commun'
       integer rgx(n), rgy(n), antirgx(n), antirgy(n),antirgt(n),
     *       ind,nstar,i,j
       real*8 xvec(n),yvec(n), xord(n), yord(n),pas,tord(n)
       real*8 Astarn(d),t(d),B,e(n)
       real*8 sum

       real*8 taut7,taus,integ(d)
       real*8 Q,z1(n),z2(n)
       do 199 ind=1,d
             t(ind)=dble(ind)/dble(d)
199          continue
       pas=t(2)-t(1)
       nstar=50.d0
       call indexx(n,xvec,antirgx)
       call indexx(n,yvec,antirgy)
       do 301 i=1,n
             xord(i)=xvec(antirgx(i))
             yord(i)=yvec(antirgy(i))
301          continue
      call rank(n,antirgx,rgx)
      call rank(n,antirgy,rgy)
      do 302 i=1,n
             z1(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
             z2(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
302    continue
       do 305 ind=1,d-1
              B=(1-t(ind))/t(ind)
       do 306 i=1,n
       Q=B*z2(i)
              if (z1(i).GT.Q) then
```

```
                        e(i)=z1(i)/n
                else
                        e(i)=Q/n
                endif
 106    enddo
        call indexx(n,e,antirgt)
        do 103 i=1,n
                tord(i)=e(antirgt(i))
 103    continue
        sum=0.d0
        j=50.d0
        do 104 i=2451,2450+35
        sum=tord(i)*j+sum
        j=j-1.d0
 104    enddo
        Astarn(ind)=t(ind)*sum/35
 105    enddo
        call tauK(t,pas,Astarn,taut7,taus,integ)
        end
*****************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE JOE ET AL. AVEC UN SEUIL DE 25/2500
*****************************************************************************
        subroutine joe200(xvec,yvec,taut7,Astarn)
        include 'commun'
        integer rgx(n), rgy(n), antirgx(n), antirgy(n),antirgt(n),
     *      ind,nstar,i,j
        real*8 xvec(n),yvec(n), xord(n), yord(n),pas,tord(n)
        real*8 Astarn(d),t(d),B,e(n)
        real*8 sum

        real*8 taut7,taus,integ(d)
        real*8 Q,z1(n),z2(n)
        do 199 ind=1,d
                t(ind)=dble(ind)/dble(d)
 199            continue
        pas=t(2)-t(1)
        nstar=200.d0
        call indexx(n,xvec,antirgx)
        call indexx(n,yvec,antirgy)
        do 301 i=1,n
                xord(i)=xvec(antirgx(i))
                yord(i)=yvec(antirgy(i))
 301            continue
      call rank(n,antirgx,rgx)
      call rank(n,antirgy,rgy)
      do 302 i=1,n
                z1(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
                z2(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
 302    continue
        do 305 ind=1,d-1
                B=(1-t(ind))/t(ind)
        do 306 i=1,n
        Q=B*z2(i)
                if (z1(i).GT.Q) then
                        e(i)=z1(i)/n
                else
                        e(i)=Q/n
                endif
 306    enddo
        call indexx(n,e,antirgt)
        do 303 i=1,n
                tord(i)=e(antirgt(i))
 303    continue
        sum=0.d0
        j=200.d0
        do 304 i=2301,2300+185
        sum=tord(i)*j+sum
        j=j-1.d0
 304    enddo
        Astarn(ind)=t(ind)*sum/185
 305    enddo
        call tauK(t,pas,Astarn,taut7,taus,integ)
        end
*****************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 25/2500
*****************************************************************************
        subroutine kho25(xvec,yvec,taut3,Astarn)
        include 'commun'
        integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *      istar(n),k,ind,nstar,i
        real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     *      xstar(n), ystar(n), rvec(n), wvec(n),rstar21
        real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)


        real*8 taut3,taus,integ(d)
        do 199 ind=1,d
                t(ind)=dble(ind)/dble(d)
 199            continue
        pas=t(2)-t(1)
        rstar21=25.d0
        nstar=0.d0
        call indexx(n,xvec,antirgx)
        call indexx(n,yvec,antirgy)
        do 101 i=1,n
                xord(i)=xvec(antirgx(i))
                yord(i)=yvec(antirgy(i))
 101            continue
        call rank(n,antirgx,rgx)
        call rank(n,antirgy,rgy)
        do i=1,n
                  rvec(i)=0.d0
        enddo
        do 102 i=1,n
                xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
                ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
                rvec(i)=xstar(i)+ystar(i)
                if (rvec(i).gt.rstar21) then
                                                nstar=nstar+1
                                                istar(nstar)=i
                                        endif
 102            continue
      do 103 k=1,nstar
                wvec(k)=xstar(istar(k))/rvec(istar(k))
 103            continue
      do 104 ind=1,d-1
                Anaux=0.d0
                do 105 k=1,nstar
                        auxil1=wvec(k)*(t(ind))
                        auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
                        if (auxil1.ge.auxil2) then
                             Anaux=Anaux+auxil1
                        else
                             Anaux=Anaux+auxil2
                        endif
 105                    continue
                Astarn(ind)=2.d0*Anaux/dble(nstar)
 104            continue
        call tauK(t,pas,Astarn,taut3,taus,integ)
        end
*****************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 100/2500
*****************************************************************************
        subroutine kho100(xvec,yvec,taut2,Astarn)
        include 'commun'
        integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *      istar(n),k,ind,nstar,i
        real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     *      xstar(n), ystar(n), rvec(n), wvec(n),rstar21
        real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)


        real*8 taut2,taus,integ(d)
        do 199 ind=1,d
                t(ind)=dble(ind)/dble(d)
 199            continue
        pas=t(2)-t(1)
        rstar21=100.d0
        nstar=0.d0
        call indexx(n,xvec,antirgx)
        call indexx(n,yvec,antirgy)
        do 101 i=1,n
                xord(i)=xvec(antirgx(i))
                yord(i)=yvec(antirgy(i))
 101            continue
      call rank(n,antirgx,rgx)
      call rank(n,antirgy,rgy)
      do i=1,n
                rvec(i)=0.d0
      enddo
      do 102 i=1,n
                xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
                ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
                rvec(i)=xstar(i)+ystar(i)
                if (rvec(i).gt.rstar21) then
                                                nstar=nstar+1
                                                istar(nstar)=i
                                        endif
 102            continue
      do 103 k=1,nstar
                wvec(k)=xstar(istar(k))/rvec(istar(k))
 103            continue
      do 104 ind=1,d-1
                Anaux=0.d0
                do 105 k=1,nstar
                        auxil1=wvec(k)*(t(ind))
                        auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
                        if (auxil1.ge.auxil2) then
                             Anaux=Anaux+auxil1
                        else
                             Anaux=Anaux+auxil2
                        endif
 105                    continue
                Astarn(ind)=2.d0*Anaux/dble(nstar)
 104            continue
        call tauK(t,pas,Astarn,taut2,taus,integ)
        end
*****************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 200/2500
*****************************************************************************
        subroutine kho200(xvec,yvec,taut1,Astarn)
        include 'commun'
        integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *      istar(n),k,ind,nstar,i

        real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     -      xstar(n), ystar(n), rvec(n), wvec(n),rstar21
        real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)

        real*8 taut1,taus,integ(d)
        do 199 ind=1,d
                t(ind)=dble(ind)/dble(d)
 199            continue
        pas=t(2)-t(1)
        rstar21=200.d0
        nstar=0.d0
        call indexx(n,xvec,antirgx)
        call indexx(n,yvec,antirgy)
        do 101 i=1,n
                xord(i)=xvec(antirgx(i))
                yord(i)=yvec(antirgy(i))
 101            continue
        call rank(n,antirgx,rgx)
        call rank(n,antirgy,rgy)
        do i=1,n
                rvec(i)=0.d0
        enddo
        do 102 i=1,n
                xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
                ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
                rvec(i)=xstar(i)+ystar(i)
                if (rvec(i).gt.rstar21) then
                                                nstar=nstar+1
                                                istar(nstar)=i
                                        endif
 102            continue
      do 103 k=1,nstar
                wvec(k)=xstar(istar(k))/rvec(istar(k))
 103            continue
      do 104 ind=1,d-1
                Anaux=0.d0
                do 105 k=1,nstar
                        auxil1=wvec(k)*(t(ind))
                        auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
                        if (auxil1.ge.auxil2) then
                             Anaux=Anaux+auxil1
                        else
                             Anaux=Anaux+auxil2
                        endif
 105                    continue
                Astarn(ind)=2.d0*Anaux/dble(nstar)
 104            continue
        call tauK(t,pas,Astarn,taut1,taus,integ)
        end
*****************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE CAPERAA ET AL. ***************************
*****************************************************
        subroutine CFG25(xvec,yvec,taut5,Anrg)
        parameter (nn=25,d=100)
        integer rgx(nn), rgy(nn), antirgx(nn), antirgy(nn),
     *      i,j,l,m
        real*8 transfx(nn),transfy(nn)
        real*8 xvec(nn),yvec(nn)
        real*8 t(d)
        real*8 maxxvec(nn),maxyvec(nn)

        real*8 Anrg(d)
        real*8 taut5
        do m=1,25
                maxxvec(m)=-99999.d0
                maxyvec(m)=-99999.d0
        enddo
        j=1
        m=1
 500    do l=j,j+99
                if (xvec(l).GE.maxxvec(m)) then
                        maxxvec(m)=xvec(l)
                else
                        maxxvec(m)=maxxvec(m)
                endif
                if (yvec(l).GE.maxyvec(m)) then
                        maxyvec(m)=yvec(l)
                else
                        maxyvec(m)=maxyvec(m)
                endif
        enddo
        j=j+100
                m=m+1
                if (m.LT.26) then
                        goto 500
                endif
        call indexx(nn,maxxvec,antirgx)
        call indexx(nn,maxyvec,antirgy)
        call rank(nn,antirgx,rgx)
        call rank(nn,antirgy,rgy)
        do 11 i=1,nn
          transfx(i)=dble(rgx(i))/dble(nn+1)
          transfy(i)=dble(rgy(i))/dble(nn+1)
 11       continue
        call subCFG2(transfx,transfy,t,Anrg,taut5)
        end
*****************************************************************************
PROCEDURE DE CALCUL DE L'ESTIMATEUR NON PARAMETRIQUE An DE LA FONCTION DE
 DEPENDANCE D'UNE COPULE DES VALEURS EXTREMES, A PARTIR D'UN ECHANTILLON (u,v)
 DE TAILLE n DE COUPLES DE MAXIMA DE MARGES UNIFORMES.
*****************************************************************************
      subroutine subCFG2(u,v,x,Acfg,taut)
      integer nn, d
      parameter (nn=25, d=100)
      integer i, ind, ind1, antirgx(nn), k2
```

```
      real*8 somv, somm, aux, pas, taut, taus
      real*8 x(d), auxAcfg(d), Acfg(d), integ(d), pl(d),
     *      logA0(d), logA1(d)
      real*8 u(nn), v(nn), z(nn), zord(nn)
      do 199 indi=1,d
            x(indi)=dble(indi)/dble(d)
199   continue
      pas=x(2)-x(1)
      do 10 indi=1,d
            pl(indi)=1-x(indi)
10    continue
      somm=0.0D0
      somv=0.0D0
      do 12 indi=1,d
            auxAcfg(indi)=1.0D0
            Acfg(indi)=0.0D0
 12     continue
      do 13 i=1,nn
            z(i)=log(u(i))/(log(u(i)*v(i)))
 13   continue
      call indexx(nn,z,antirgz)
      do 77 i=1,nn
        zord(i)=z(antirgz(i))
77    continue
       do ind=1,d
                logA0(ind)=0.d0
                logA1(ind)=0.d0
                Acfg(ind)=0.d0
       enddo
     ind=1
700  if (x(ind) le.z(antirgz(1))) then
                logA0(ind)= log(1-x(ind))
                do 37 i=1,nn
                   logA1(ind)=logA1(ind)+log(z(antirgz(i)))-
     *                        log(1-z(antirgz(i)))
37              continue
                logA1(ind)=logA1(ind)/dble(nn) + log(1-x(ind))

                Acfg(ind) = exp(pl(ind)*logA0(ind)+
     *                      (1-pl(ind))*logA1(ind))
                ind=ind+1
     goto 700
     endif
     k2=1
     aux=z(antirgz(nn))
800  if (x(ind).le.aux) then
801      if (x(ind).le.z(antirgz(k2+1))) then
              do 26 i=1,k2
                   logA0(ind)=logA0(ind)+log(1-z(antirgz(i)))
     *                        -log(z(antirgz(i)))
26            continue
              logA0(ind)=logA0(ind)/dble(nn)
     *                  + dble(k2)*log(x(ind))/dble(nn)
     *                  + (1-dble(k2)/dble(nn))*log(1-x(ind))
              do 27 i=k2+1,nn
                   logA1(ind)=logA1(ind)+log(z(antirgz(i)))
     *                        -log(1-z(antirgz(i)))
27            continue
              logA1(ind)=logA1(ind)/dble(nn)
     *                  +dble(k2)*log(x(ind))/dble(nn)
     *                  +(1-dble(k2)/dble(nn))*log(1-x(ind))
              Acfg(ind)=exp(pl(ind)*logA0(ind)
     *                  + (1-pl(ind))*logA1(ind))
              ind=ind+1
              goto 801
         endif
         k2=k2+1
         goto 800
     endif
900  if (ind.le.d) then
        do 28 i=1,nn
              logA0(ind)=logA0(ind)+log(1-z(antirgz(i)))-
     *                   log(z(antirgz(i)))
28      continue
         logA0(ind)=logA0(ind)/dble(nn) + log(x(ind))
         logA1(ind)=log(x(ind))
         Acfg(ind)=exp(pl(ind)*logA0(ind)+(1-pl(ind))*logA1(ind))
         ind=ind+1
        goto 900
     endif
       call tauK(x,pas,Acfg,taut,taus,integ)
      end
*************************************************************************
PROCEDURE CALCULANT L'ERREUR L1 ENTRE LA FONCTION A MIXTE (A3) AVEC
 THETA=0.668 ET L'ESTIMATION Astarnl.
*************************************************************************
        subroutine errl1(Astarnl,erreurL)
        include 'commun'
        integer ind,q
        parameter(q=100)
        integer l
        real*8 Astarnbar(d),Astarnl(d,d),erreurL1(d),j,erreurL(d)
        j=0.01d0
        do ind=1,d-1
                Astarnbar(ind)=0.668d0*(j)**2-0.668d0*(j)+1.d0
                j=j+0.01d0
        enddo
        do l=1,q
        erreurL1(l)=0.d0
        enddo
        do l=1,q
        do ind=1,d-1
                erreurL1(l)=erreurL1(l)+abs(Astarnbar(ind)-Astarnl(ind,l))
        enddo
        enddo
        do l=1,q
        erreurL(l)=erreurL1(l)*0.01d0
        enddo
        end
*************************************************************************
PROCEDURE CALCULANT L'ERREUR L1 ENTRE LA FONCTION ASYMETRIQUE (A4) (Astarnbar)
   ET L'ESTIMATION Astarnl.
*************************************************************************
        subroutine errl2(Astarnl,erreurL)
        include 'commun'
        integer ind,q
        parameter(q=100)
        integer l
        real*8 Astarnbar(d),Astarnl(d,d),erreurL1(d),j,erreurL(d)
        j=0.01d0
        do ind=1,d-1
                if (j.lt.0.3333d0) then
                        Astarnbar(ind)=1.5d0*(j)**2-j+1.d0
                else
                        Astarnbar(ind)=0.125d0*(3*(j)**2-2*j+7)
                endif
                j=j+0.01d0
        enddo
        do l=1,q
        erreurL1(l)=0.d0
        enddo
        do l=1,q
        do ind=1,d-1
                erreurL1(l)=erreurL1(l)+abs(Astarnbar(ind)-Astarnl(ind,l))

        enddo
        enddo
        do l=1,q
        erreurL(l)=erreurL1(l)*0.01d0
        enddo
        end
*************************************************************************
PROCEDURE CALCULANT L'ERREUR L1 ENTRE LA FONCTION A MIXTE (A1) AVEC
 THETA=0.3 ET L'ESTIMATION Astarnl.
*************************************************************************
        subroutine errl12(Astarnl,erreurL)
        include 'commun'
        integer ind,q
        parameter(q=100)
        integer l
        real*8 Astarnbar(d),Astarnl(d,d),erreurL1(d),j,erreurL(d)
        j=0.01d0
        do ind=1,d-1
                Astarnbar(ind)=0.3d0*(j)**2-0.3d0*(j)+1.d0
                j=j+0.01d0
        enddo
        do l=1,q
        erreurL1(l)=0.d0
        enddo
        do l=1,q
        do ind=1,d-1
                erreurL1(l)=erreurL1(l)+abs(Astarnbar(ind)-Astarnl(ind,l))
        enddo
        enddo
        do l=1,q
        erreurL(l)=erreurL1(l)*0.01d0
        enddo
        end
*************************************************************************
PROCEDURE CALCULANT L'ERREUR L1 ENTRE LA FONCTION ASYMETRIQUE (A2)    ET
  L'ESTIMATION Astarnl.
*************************************************************************

        subroutine errl3(Astarnl,erreurL)
        include 'commun'
        integer ind,q
        parameter(q=100)
        integer l
        real*8 Astarnbar(d),Astarnl(d,d),erreurL1(d),j,erreurL(d)
        j=0.01d0
        do ind=1,d-1
                if (j.lE. 0.2d0) then
                        Astarnbar(ind)=1.875d0*(j)**2-0.75d0*j+1.d0
                else
                        Astarnbar(ind)=0.0078125d0*(15*(j)**2-6*j+119)
                endif
                j=j+0.01d0
        enddo
        do l=1,q
        erreurL1(l)=0.d0
        enddo
        do l=1,q
        do ind=1,d-1
                erreurL1(l)=erreurL1(l)+abs(Astarnbar(ind)-Astarnl(ind,l))
        enddo
        enddo
        do l=1,q
        erreurL(l)=erreurL1(l)*0.01d0
        enddo
        end
*************************************************************************
FICHIER DE DECLARATIONS DE VARIABLES COMMUNES AUX DIFFERENTS PROGRAMMES ET
SOUS-PROGRAMMES
*************************************************************************

       implicit none
       integer n, d, nrepli
       parameter (n=2500, d=100, nrepli=50)
       real*8 r, alpha, m, parA, c, ord
       real*8 drand
       common /var1/ r
       common /var2/ m, alpha, c, ord
       common /var3/ parA
       external drand
```

Annexe G

Programmes informatiques de fortran 77 utilisés pour l'analyse 2

```
*************************************************************************
PROGRAMME PRINCIPAL CALCULANT LES VALEURS DU TAU DE KENDALL ET DE L'ERREUR L1
POUR DES ESTIMATIONS DE LA FONCTION A PAR LES METHODES DE JOE ET AL., DE
 KHOUDRAJI, ET DE CAPERAA ET AL., A PARTIR D'ECHANTILLONS DE TAILLE 50 ou 100 DE
 COPULE DE GUMBEL B OU DE COPULE ASYMETRIQUE DE TAWN
*************************************************************************
        program analyseA3
        integer n,d,q,l,k,j
        parameter (n=100,nj=100,d=100,q=100)
        real*8 u(n),v(n),depart,A(d),tau(d,d),drand,A2(d)
        real*8 A1(d,d),erreurL11(d,d),erreurL1(d),tau1(d,d)
        real*8 EQM(d),errL1bar(d),errL1car(d),taubar(d)
        external drand
        depart=drand(1212)
       open(3,file='suma3')
       open(2,file='resA3sum')
        do l=1,q
                j=1
                call gumsym1(depart,u,v)
                call kho505(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS1(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=2
                call gumsym1(depart,u,v)
                call kho5010(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS1(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=3
                call gumsym1(depart,u,v)
                call joe505(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS1(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=4
                call gumsym1(depart,u,v)
                call joe5010(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS1(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=5
                call gumsym1(depart,u,v)
                call CFG1(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS1(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=6
                call gumsym2(depart,u,v)
                call kho505(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS2(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=7
                call gumsym2(depart,u,v)
                call kho5010(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS2(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=8
                call gumsym2(depart,u,v)
                call joe505(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS2(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=9
                call gumsym2(depart,u,v)
                call joe5010(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS2(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=10
                call gumsym2(depart,u,v)
                call CFG1(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS2(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=11
                call gumsym3(depart,u,v)
                call kho505(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS3(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=12
                call gumsym3(depart,u,v)
                call kho5010(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS3(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=13
                call gumsym3(depart,u,v)
                call joe505(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS3(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=14
                call gumsym3(depart,u,v)
                call joe5010(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS3(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=15
                call gumsym3(depart,u,v)
                call CFG1(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        do ind=1,d-1
                A2(ind)=0.d0
        enddo
        call errlGS3(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=16
                call gumsym4(depart,u,v)
                call kho10010(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS1(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=17
                call gumsym4(depart,u,v)
                call kho10020(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS1(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=18
                call gumsym4(depart,u,v)
                call joe10010(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS1(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=19
                call gumsym4(depart,u,v)
                call joe10020(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS1(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=20
                call gumsym4(depart,u,v)
                call CFG2(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS1(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=21
                call gumsym5(depart,u,v)
                call kho10010(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS2(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=22
                call gumsym5(depart,u,v)
                call kho10020(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS2(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=23
                call gumsym5(depart,u,v)
                call joe10010(u,v,tau(j,l),A)
        do ind=1,d-1
                A1(ind,l)=A(ind)
        enddo
        enddo
        call errlGS2(A1,erreurL1)
        do l=1,q
                erreurL11(j,l)=erreurL1(l)
        enddo
        do l=1,q
                j=24
                call gumsym5(depart,u,v)
                call joe10020(u,v,tau(j,l),A)
        do ind=1,d-1
```

```
        A1(ind,l)=A(ind)
enddo
enddo
call errlGS2(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=25
        call gumsym5(depart,u,v)
        call CFG2(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGS2(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=26
        call gumsym6(depart,u,v)
        call kho10010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=27
        call gumsym6(depart,u,v)
        call kho10020(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=28
        call gumsym6(depart,u,v)
        call joe10010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=29
        call gumsym6(depart,u,v)
        call joe10020(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=30
        call gumsym6(depart,u,v)
        call CFG2(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=31
        call gumasym1(depart,u,v)
        call kho505(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS1(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=32
        call gumasym1(depart,u,v)
        call kho5010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS1(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=33
        call gumasym1(depart,u,v)
        call joe505(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS1(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=34
        call gumasym1(depart,u,v)
        call joe5010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS1(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=35
        call gumasym1(depart,u,v)
        call CFG1(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS1(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=36
        call gumasym2(depart,u,v)
        call kho505(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS2(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=37
        call gumasym2(depart,u,v)
        call kho5010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS2(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=38
        call gumasym2(depart,u,v)
        call joe505(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS2(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=39
        call gumasym2(depart,u,v)
        call joe5010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS2(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=40
        call gumasym2(depart,u,v)
        call CFG1(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS2(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=41
        call gumasym3(depart,u,v)
        call kho505(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=42
        call gumasym3(depart,u,v)
        call kho5010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=43
        call gumasym3(depart,u,v)
        call joe505(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=44
        call gumasym3(depart,u,v)
        call joe5010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS3(A1,erreurL1)

do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=45
        call gumasym3(depart,u,v)
        call CFG1(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=46
        call gumasym4(depart,u,v)
        call kho10010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS1(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=47
        call gumasym4(depart,u,v)
        call kho10020(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS1(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=48
        call gumasym4(depart,u,v)
        call joe10010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS1(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=49
        call gumasym4(depart,u,v)
        call joe10020(u,v,tau(j,l),A)
```

```
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS1(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=50
        call gumasym4(depart,u,v)
        call CFG2(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS1(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=51
        call gumasym5(depart,u,v)
        call kho10010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS2(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=52
        call gumasym5(depart,u,v)
        call kho10020(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS2(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=53
        call gumasym5(depart,u,v)
        call joe10010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS2(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=54
        call gumasym5(depart,u,v)
        call joe10020(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS2(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=55
        call gumasym5(depart,u,v)
        call CFG2(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS2(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=56
        call gumasym6(depart,u,v)
        call kho10010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=57
        call gumasym6(depart,u,v)
        call kho10020(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=58
        call gumasym6(depart,u,v)
        call joe10010(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=59
        call gumasym6(depart,u,v)
        call joe10020(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do l=1,q
        j=60
        call gumasym6(depart,u,v)
        call CFG2(u,v,tau(j,l),A)
do ind=1,d-1
        A1(ind,l)=A(ind)
enddo
enddo
call errlGAS3(A1,erreurL1)
do l=1,q
        erreurL11(j,l)=erreurL1(l)
enddo
do j=1,60
        errL1bar(j)=0.d0
        errL1car(j)=0.d0
        taubar(j)=0.d0
        EQM(j)=0.d0
enddo
k=1
do i=1,4
do j=k,k+4
        do l=1,q
                errL1bar(j)=errL1bar(j)+(erreurL11(j,l))/q
                errL1car(j)=errL1car(j)+(erreurL11(j,l)**2)
                taubar(j)=taubar(j)+(tau(j,l))/q
                EQM(j)=EQM(j)+((tau(j,l)-0.25)**2)/q
        enddo
enddo
        k=k+15
enddo
k=6
do i=1,4
do j=k,k+4
        do l=1,q
                errL1bar(j)=errL1bar(j)+(erreurL11(j,l))/q
                errL1car(j)=errL1car(j)+(erreurL11(j,l)**2)
                taubar(j)=taubar(j)+(tau(j,l))/q
                EQM(j)=EQM(j)+((tau(j,l)-0.50)**2)/q
        enddo
enddo
        k=k+15
enddo
k=11
do i=1,4
do j=k,k+4
        do l=1,q
                errL1bar(j)=errL1bar(j)+(erreurL11(j,l))/q
                errL1car(j)=errL1car(j)+(erreurL11(j,l)**2)
                taubar(j)=taubar(j)+(tau(j,l))/q
                EQM(j)=EQM(j)+((tau(j,l)-0.75)**2)/q
        enddo
enddo
        k=k+15
enddo
do j=1,60
        print*,j,errL1bar(j),errL1car(j),taubar(j),EQM(j)
enddo
        do j=1,60
        write(2,1003) taubar(j),EQM(j),errL1bar(j),errL1car(j)
1003    format(4(1X,F17.12),/)
        enddo
        do j=1,60
                do l=1,q
                        write(3,1006) tau(j,l),erreurL11(j,l),taul(j,l)
1006                    format(1X,F17.12,1X,F17.12,1X,F17.12)

                enddo
        enddo

      close(3)
      close(2)
      end
**************************************************************************
PROCEDURE GENERANT 50 COUPLES (u,v) DE COPULE DE GUMBEL DE TYPE B AVEC r=4/3
**************************************************************************
      subroutine gumsym1(depart,u,v)
      integer n, d, m, k, kasym
      parameter (n=50, d=100, m=500, k=1, kasym=3)
      integer i
      real*8 depart, uni1,uni2,uni3,uni4,uni5,auxgumb1,auxgumb2,r,
     *       pt, drand, rand, AfoncGB, AfoncGBasym
      real*8 u(n), v(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external drand,rand
      external lgvraisasym, lgvrais
      external phi, D01AMF
      external AfoncGB, AfoncGBasym
      r=1.333333D0
          do 211 i=1,n
             uni1=0.D0
             uni2=0.D0
             uni3=0.D0
             uni4=0.D0
             uni5=0.D0
 112         if ((uni1.LE.0.D0).OR.(uni2.LE.0.D0).
     *         OR.(uni3.LE.0.D0).OR.(uni4.LE.0.d0).
     *         OR.(uni5.LE.0.D0).OR.(uni1.GE.1.D0).OR.(uni2.GE.1.D0).
     *         OR.(uni3.GE.1.D0).OR.(uni4.GE.1.d0).OR.(uni5.GE.1.D0).EQ.1) the
n
             uni1=drand(0)
             uni2=drand(0)
             uni3=drand(0)
             uni4=drand(0)
             uni5=drand(0)
             goto 112
             endif
             auxgumb1=uni5**(1/r)/(uni5**(1/r)+(1-uni5)**(1/r))
              if(uni4.le.1/r) then
                             auxgumb2=uni1*uni2
                             else
                             auxgumb2=uni3
             endif
             u(i)=auxgumb2**(auxgumb1/AfoncGB(auxgumb1))
             v(i)=auxgumb2**((1-auxgumb1)/AfoncGB(auxgumb1))
211        continue
      end
**************************************************************************
PROCEDURE GENERANT 50 COUPLES (u,v) DE COPULE DE GUMBEL DE TYPE B AVEC r=2
**************************************************************************
      subroutine gumsym2(depart,u,v)
      integer n, d, m, k, kasym
      parameter (n=50, d=100, m=500, k=1, kasym=1)
      integer i
      real*8 depart,
     *       uni1,uni2,uni3,uni4,uni5,auxgumb1,auxgumb2,r,
     *       pt, drand, rand,
     *       AfoncGB, AfoncGBasym
      real*8 u(n), v(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external drand,rand
      external lgvraisasym, lgvrais
      external phi, D01AMF
      external AfoncGB, AfoncGBasym
       r=2.D0
          do 211 i=1,n
             uni1=0.D0
             uni2=0.D0
             uni3=0.D0
             uni4=0.D0
             uni5=0.D0
 112         if ((uni1.LE.0.D0).OR.(uni2.LE.0.D0).
     *         OR.(uni3.LE.0.D0).OR.(uni4.LE.0.d0).
     *         OR.(uni5.LE.0.D0).OR.(uni1.GE.1.D0).
     *         OR.(uni2.GE.1.D0).
     *         OR.(uni3.GE.1.D0).OR.(uni4.GE.1.d0).
     *         OR.(uni5.GE.1.D0).EQ.1) then
             uni1=drand(0)
             uni2=drand(0)
             uni3=drand(0)
             uni4=drand(0)
             uni5=drand(0)
             goto 112
             endif
            auxgumb1=uni5**(1/r)/(uni5**(1/r)+(1-uni5)**(1/r))
              if(uni4.le.1/r) then
                             auxgumb2=uni1*uni2
                             else
                             auxgumb2=uni3
             endif
             u(i)=auxgumb2**(auxgumb1/AfoncGB(auxgumb1))
             v(i)=auxgumb2**((1-auxgumb1)/AfoncGB(auxgumb1))
211        continue
      end
**************************************************************************
PROCEDURE GENERANT 50 COUPLES (u,v) DE COPULE DE GUMBEL DE TYPE B AVEC r=4
**************************************************************************
      subroutine gumsym3(depart,u,v)
      integer n, d, m, k, kasym
```

```
      parameter (n=50, d=100, m=500, k=1, kasym=3)
      integer i
      real*8 depart,
     *        uni1,uni2,uni3,uni4,uni5,auxgumb1,auxgumb2,r,
     *        pt, drand, rand,
     *        AfoncGB, AfoncGBasym
      real*8 u(n), v(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external drand,rand
      external lgvraisasym, lgvrais
      external phi, D01AMF
      external AfoncGB, AfoncGBasym
       r=4.D0
            do 211 i=1,n
               uni1=0.D0
               uni2=0.D0
               uni3=0.D0
               uni4=0.D0
               uni5=0.D0
112            if ((uni1.LE.0.D0).OR.(uni2.LE.0.D0).
     *           OR.(uni3.LE.0.D0).OR.(uni4.LE.0.d0).
     *           OR.(uni5.LE.0.D0).OR.(uni1.GE.1.D0).
     *           OR.(uni2.GE.1.D0).
     *           OR.(uni3.GE.1.D0).OR.(uni4.GE.1.d0).
     *           OR.(uni5.GE.1.D0).EQ.1) then
               uni1=drand(0)
               uni2=drand(0)
               uni3=drand(0)
               uni4=drand(0)
               uni5=drand(0)
               goto 112
               endif
               auxgumb1=uni5**(1/r)/(uni5**(1/r)+(1-uni5)**(1/r))
                if(uni4.le.1/r) then
                               auxgumb2=uni1*uni2
                               else
                               auxgumb2=uni3
               endif
               u(i)=auxgumb2**(auxgumb1/AfoncGB(auxgumb1))
               v(i)=auxgumb2**((1-auxgumb1)/AfoncGB(auxgumb1))
211         continue
         end
**********************************************************************
PROCEDURE GENERANT 100 COUPLES (u,v) DE COPULE DE GUMBEL DE TYPE B AVEC r=4/3
**********************************************************************
         subroutine gumsym4(depart,u,v)
             integer n, d, m, k, kasym
      parameter (n=100, d=100, m=500, k=1, kasym=3)
      integer i
      real*8 depart,
     *        uni1,uni2,uni3,uni4,uni5,auxgumb1,auxgumb2,r,
     *        pt, drand, rand,
     *        AfoncGB, AfoncGBasym
      real*8 u(n), v(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external drand,rand
      external lgvraisasym, lgvrais
      external phi, D01AMF
      external AfoncGB, AfoncGBasym
       r=1.333333333333D0
            do 211 i=1,n
               uni1=0.D0
               uni2=0.D0
               uni3=0.D0
               uni4=0.D0
               uni5=0.D0
112            if ((uni1.LE.0.D0).OR.(uni2.LE.0.D0).
     *           OR.(uni3.LE.0.D0).OR.(uni4.LE.0.d0).
     *           OR.(uni5.LE.0.D0).OR.(uni1.GE.1.D0).
     *           OR.(uni2.GE.1.D0).
     *           OR.(uni3.GE.1.D0).OR.(uni4.GE.1.d0).
     *           OR.(uni5.GE.1.D0).EQ.1) then
               uni1=drand(0)
               uni2=drand(0)
               uni3=drand(0)
               uni4=drand(0)
               uni5=drand(0)
               goto 112
               endif
               auxgumb1=uni5**(1/r)/(uni5**(1/r)+(1-uni5)**(1/r))
                if(uni4.le.1/r) then
                               auxgumb2=uni1*uni2
                               else
                               auxgumb2=uni3
               endif
               u(i)=auxgumb2**(auxgumb1/AfoncGB(auxgumb1))
               v(i)=auxgumb2**((1-auxgumb1)/AfoncGB(auxgumb1))
211         continue
         end
**********************************************************************
PROCEDURE GENERANT 100 COUPLES (u,v) DE COPULE DE GUMBEL DE TYPE B AVEC r=2
**********************************************************************
      subroutine gumsym5(depart,u,v)
      integer n, d, m, k, kasym
      parameter (n=100, d=100, m=500, k=1, kasym=3)
      integer i
      real*8 depart,
     *        uni1,uni2,uni3,uni4,uni5,auxgumb1,auxgumb2,r,
     *        pt, drand, rand,
     *        AfoncGB, AfoncGBasym
      real*8 u(n), v(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external drand,rand
      external lgvraisasym, lgvrais
      external phi, D01AMF
      external AfoncGB, AfoncGBasym
       r=2.D0
            do 211 i=1,n
               uni1=0.D0
               uni2=0.D0
               uni3=0.D0
               uni4=0.D0
               uni5=0.D0
112            if ((uni1.LE.0.D0).OR.(uni2.LE.0.D0).
     *           OR.(uni3.LE.0.D0).OR.(uni4.LE.0.d0).
     *           OR.(uni5.LE.0.D0).OR.(uni1.GE.1.D0).
     *           OR.(uni2.GE.1.D0).
     *           OR.(uni3.GE.1.D0).OR.(uni4.GE.1.d0).
     *           OR.(uni5.GE.1.D0).EQ.1) then
               uni1=drand(0)
               uni2=drand(0)
               uni3=drand(0)
               uni4=drand(0)
               uni5=drand(0)
               goto 112
               endif
               auxgumb1=uni5**(1/r)/(uni5**(1/r)+(1-uni5)**(1/r))
                if(uni4.le.1/r) then
                               auxgumb2=uni1*uni2
                               else
                               auxgumb2=uni3
               endif
               u(i)=auxgumb2**(auxgumb1/AfoncGB(auxgumb1))
               v(i)=auxgumb2**((1-auxgumb1)/AfoncGB(auxgumb1))
211         continue
         end
**********************************************************************
PROCEDURE GENERANT 100 COUPLES (u,v) DE COPULE DE GUMBEL DE TYPE B AVEC r=4
**********************************************************************
      subroutine gumsym6(depart,u,v)
      integer n, d, m, k, kasym
      parameter (n=100, d=100, m=500, k=1, kasym=3)
      integer i
      real*8 depart,
     *        uni1,uni2,uni3,uni4,uni5,auxgumb1,auxgumb2,r,
     *        pt, drand, rand,
     *        AfoncGB, AfoncGBasym
      real*8 u(n), v(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external drand,rand
      external lgvraisasym, lgvrais
      external phi, D01AMF
      external AfoncGB, AfoncGBasym
       r=4.D0
            do 211 i=1,n
               uni1=0.D0
               uni2=0.D0
               uni3=0.D0
               uni4=0.D0
               uni5=0.D0
112            if ((uni1.LE.0.D0).OR.(uni2.LE.0.D0).
     *           OR.(uni3.LE.0.D0).OR.(uni4.LE.0.d0).
     *           OR.(uni5.LE.0.D0).OR.(uni1.GE.1.D0).
     *           OR.(uni2.GE.1.D0).
     *           OR.(uni3.GE.1.D0).OR.(uni4.GE.1.d0).
     *           OR.(uni5.GE.1.D0).EQ.1) then
               uni1=drand(0)
               uni2=drand(0)
               uni3=drand(0)
               uni4=drand(0)
               uni5=drand(0)
               goto 112
               endif
               auxgumb1=uni5**(1/r)/(uni5**(1/r)+(1-uni5)**(1/r))
                if(uni4.le.1/r) then
                               auxgumb2=uni1*uni2
                               else
                               auxgumb2=uni3
               endif
               u(i)=auxgumb2**(auxgumb1/AfoncGB(auxgumb1))
               v(i)=auxgumb2**((1-auxgumb1)/AfoncGB(auxgumb1))
211         continue
         end
**********************************************************************
PROCEDURE GENERANT 50 COUPLES (u,v) DE COPULE ASYMETRIQUE DE TAWN AVEC
 alpha=0.78, beta=0.97 et r=1.42
**********************************************************************
      subroutine gumasym1(depart,u,v)
      integer n, d, m, k, kasym
      parameter (n=50, d=100, m=500, k=1, kasym=3)
      integer i
      real*8 depart,
     *        uni1,uni2,uni3,uni4,uni5,auxgumb1,auxgumb2,r,
     *        pt, drand, rand,ugumbel,vgumbel,uindep,vindep,
     *        AfoncGB, AfoncGBasym,al,be
      real*8 u(n), v(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external drand,rand
      external lgvraisasym, lgvrais
      external phi, D01AMF
      external AfoncGB, AfoncGBasym
       r=1.42D0
       al=0.78d0
       be=0.97d0
            do 111 i=1,n
               uni1=0.D0
               uni2=0.D0
               uni3=0.D0
               uni4=0.D0
               uni5=0.D0
113            if((uni1.LE.0.D0).OR.(uni2.LE.0.D0).
     *           OR.(uni3.LE.0.D0).OR.(uni4.LE.0.d0).
     *           OR.(uni5.LE.0.D0).OR.(uni1.GE.1.D0).
     *           OR.(uni2.GE.1.D0).
     *           OR.(uni3.GE.1.D0).OR.(uni4.GE.1.d0).
     *           OR.(uni5.GE.1.D0)) then
               uni1=drand(0)
               uni2=drand(0)
               uni3=drand(0)
               uni4=drand(0)
               uni5=drand(0)
               goto 113
               endif
               auxgumb1=exp(log(uni5)/r)/
     *              (exp(log(uni5)/r) + exp(log(1-uni5)/r))
               if(uni4.le.1/r) then
                               auxgumb2=uni1*uni2
                               else
                               auxgumb2=uni3
               endif
               ugumbel=exp(auxgumb1*log(auxgumb2)/AfoncGB(auxgumb1))
               vgumbel=exp((1-auxgumb1)*log(auxgumb2)/AfoncGB(auxgumb1))
               uindep=0.D0
               vindep=0.D0
114            if (uindep.LE.0.D0.OR.vindep.LE.0.D0.
     *            OR.uindep.GE.1.D0.OR.vindep.GE.1.D0) then
               uindep=drand(0)
               vindep=drand(0)
               goto 114
               endif
               u(i)=max(uindep**(1/(1-al)),ugumbel**(1/al))
               v(i)=max(vindep**(1/(1-be)),vgumbel**(1/be))
111         continue
         end
**********************************************************************
PROCEDURE GENERANT 50 COUPLES (u,v) DE COPULE ASYMETRIQUE DE TAWN AVEC
 alpha=0.78, beta=0.97 et r=2.58
**********************************************************************
      subroutine gumasym2(depart,u,v)
      integer n, d, m, k, kasym
      parameter (n=50, d=100, m=500, k=1, kasym=1)
      integer i
      real*8 depart,
     *        uni1,uni2,uni3,uni4,uni5,auxgumb1,auxgumb2,r,
     *        pt, drand, rand,ugumbel,vgumbel,uindep,vindep,
     *        AfoncGB, AfoncGBasym,al,be
      real*8 u(n), v(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external drand,rand
      external lgvraisasym, lgvrais
      external phi, D01AMF
      external AfoncGB, AfoncGBasym
       r=2.58D0
       al=0.78d0
       be=0.97d0
            do 111 i=1,n
               uni1=0.D0
               uni2=0.D0
               uni3=0.D0
               uni4=0.D0
               uni5=0.D0
113            if((uni1.LE.0.D0).OR.(uni2.LE.0.D0).
     *           OR.(uni3.LE.0.D0).OR.(uni4.LE.0.d0).
     *           OR.(uni5.LE.0.D0).OR.(uni1.GE.1.D0).
     *           OR.(uni2.GE.1.D0).
     *           OR.(uni3.GE.1.D0).OR.(uni4.GE.1.d0).
     *           OR.(uni5.GE.1.D0)) then
               uni1=drand(0)
               uni2=drand(0)
               uni3=drand(0)
               uni4=drand(0)
               uni5=drand(0)
               goto 113
               endif
               auxgumb1=exp(log(uni5)/r)/
     *              (exp(log(uni5)/r) + exp(log(1-uni5)/r))
               if(uni4.le.1/r) then
                               auxgumb2=uni1*uni2
```

```
                                 else
                                 auxgumb2=uni3
                 endif
                 ugumbel=exp(auxgumb1*log(auxgumb2)/AfoncGB(auxgumb1))
                 vgumbel=exp((1-auxgumb1)*log(auxgumb2)/AfoncGB(auxgumb1))
                 uindep=0.D0
                 vindep=0.D0
 114             if (uindep.LE.0.D0.OR.vindep.LE.0.D0.
     *              OR.uindep.GE.1.D0.OR.vindep.GE.1.D0) then
                 uindep=drand(0)
                 vindep=drand(0)
                 goto 114
                 endif
                 u(i)=max(uindep**(1/(1-al)),ugumbel**(1/al))
                 v(i)=max(vindep**(1/(1-be)),vgumbel**(1/be))
111           continue
        end
****************************************************************************
PROCEDURE GENERANT 50 COUPLES (u,v) DE COPULE ASYMETRIQUE DE TAWN AVEC
 alpha=0.78, beta=0.97 et r=50
****************************************************************************
      subroutine gumasym1(depart,u,v)
      integer n, d, m, k, kasym
      parameter (n=50, d=100, m=500, k=1, kasym=3)
      integer i
      real*8 depart,
     *        uni1,uni2,uni3,uni4,uni5,auxgumb1,auxgumb2,r,
     *        pt, drand, rand,ugumbel,vgumbel,uindep,vindep,
     *        AfoncGB, AfoncGBasym,al,be
      real*8 u(n), v(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external drand,rand
      external lgvraisasym, lgvrais
      external phi, D01AMF
      external AfoncGB, AfoncGBasym
       r=50.D0
       al=0.78d0
       be=0.97d0
            do 111 i=1,n
                 uni1=0.D0
                 uni2=0.D0
                 uni3=0.D0
                 uni4=0.D0
                 uni5=0.D0
 113             if((uni1.LE.0.D0).OR.(uni2.LE.0.D0).
     *             OR.(uni3.LE.0.D0).OR.(uni4.LE.0.d0).
     *             OR.(uni5.LE.0.D0).OR.(uni1.GE.1.D0).
     *             OR.(uni2.GE.1.D0).
     *             OR.(uni3.GE.1.D0).OR.(uni4.GE.1.d0).
     *             OR.(uni5.GE.1.D0)) then
                 uni1=drand(0)
                 uni2=drand(0)
                 uni3=drand(0)
                 uni4=drand(0)
                 uni5=drand(0)
                 goto 113
                 endif
                 auxgumb1=exp(log(uni5)/r)/
     *                  (exp(log(uni5)/r) + exp(log(1-uni5)/r))
                 if(uni4.le.1/r) then
                                 auxgumb2=uni1*uni2
                                 else
                                 auxgumb2=uni3
                 endif
                 ugumbel=exp(auxgumb1*log(auxgumb2)/AfoncGB(auxgumb1))
                 vgumbel=exp((1-auxgumb1)*log(auxgumb2)/AfoncGB(auxgumb1))
                 uindep=0.D0
                 vindep=0.D0
 114             if (uindep.LE.0.D0.OR.vindep.LE.0.D0.
     *              OR.uindep.GE.1.D0.OR.vindep.GE.1.D0) then
                 uindep=drand(0)
                 vindep=drand(0)
                 goto 114
                 endif
                 u(i)=max(uindep**(1/(1-al)),ugumbel**(1/al))
                 v(i)=max(vindep**(1/(1-be)),vgumbel**(1/be))
111           continue
        end
****************************************************************************
PROCEDURE GENERANT 100 COUPLES (u,v) DE COPULE ASYMETRIQUE DE TAWN AVEC
 alpha=0.78, beta=0.97 et r=1.42
****************************************************************************
      subroutine gumasym4(depart,u,v)
      integer n, d, m, k, kasym
      parameter (n=100, d=100, m=500, k=1, kasym=3)
      integer i
      real*8 depart,
     *        uni1,uni2,uni3,uni4,uni5,auxgumb1,auxgumb2,r,
     *        pt, drand, rand,ugumbel,vgumbel,uindep,vindep,
     *        AfoncGB, AfoncGBasym,al,be
      real*8 u(n), v(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external drand,rand
      external lgvraisasym, lgvrais
      external phi, D01AMF
      external AfoncGB, AfoncGBasym
       r=1.42D0
       al=0.78d0
       be=0.97d0
            do 111 i=1,n
                 uni1=0.D0
                 uni2=0.D0
                 uni3=0.D0
                 uni4=0.D0
                 uni5=0.D0
 113             if((uni1.LE.0.D0).OR.(uni2.LE.0.D0).
     *             OR.(uni3.LE.0.D0).OR.(uni4.LE.0.d0).
     *             OR.(uni5.LE.0.D0).OR.(uni1.GE.1.D0).
     *             OR.(uni2.GE.1.D0).
     *             OR.(uni3.GE.1.D0).OR.(uni4.GE.1.d0).
     *             OR.(uni5.GE.1.D0)) then
                 uni1=drand(0)
                 uni2=drand(0)
                 uni3=drand(0)
                 uni4=drand(0)
                 uni5=drand(0)
                 goto 113
                 endif
                 auxgumb1=exp(log(uni5)/r)/
     *                  (exp(log(uni5)/r) + exp(log(1-uni5)/r))
                 if(uni4.le.1/r) then
                                 auxgumb2=uni1*uni2
                                 else
                                 auxgumb2=uni3
                 endif
                 ugumbel=exp(auxgumb1*log(auxgumb2)/AfoncGB(auxgumb1))
                 vgumbel=exp((1-auxgumb1)*log(auxgumb2)/AfoncGB(auxgumb1))
                 uindep=0.D0
                 vindep=0.D0
 114             if (uindep.LE.0.D0.OR.vindep.LE.0.D0.
     *              OR.uindep.GE.1.D0.OR.vindep.GE.1.D0) then
                 uindep=drand(0)
                 vindep=drand(0)
                 goto 114
                 endif
                 u(i)=max(uindep**(1/(1-al)),ugumbel**(1/al))
                 v(i)=max(vindep**(1/(1-be)),vgumbel**(1/be))
111           continue
        end
****************************************************************************
PROCEDURE GENERANT 100 COUPLES (u,v) DE COPULE ASYMETRIQUE DE TAWN AVEC
 alpha=0.78, beta=0.97 et r=2.58
****************************************************************************
      subroutine gumasym5(depart,u,v)
      integer n, d, m, k, kasym
      parameter (n=100, d=100, m=500, k=1, kasym=3)
      integer i
      real*8 depart,
     *        uni1,uni2,uni3,uni4,uni5,auxgumb1,auxgumb2,r,
     *        pt, drand, rand,ugumbel,vgumbel,uindep,vindep,
     *        AfoncGB, AfoncGBasym,al,be
      real*8 u(n), v(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external drand,rand
      external lgvraisasym, lgvrais
      external phi, D01AMF
      external AfoncGB, AfoncGBasym
       r=2.58D0
       al=0.78d0
       be=0.97d0
            do 111 i=1,n
                 uni1=0.D0
                 uni2=0.D0
                 uni3=0.D0
                 uni4=0.D0
                 uni5=0.D0
 113             if((uni1.LE.0.D0).OR.(uni2.LE.0.D0).
     *             OR.(uni3.LE.0.D0).OR.(uni4.LE.0.d0).
     *             OR.(uni5.LE.0.D0).OR.(uni1.GE.1.D0).
     *             OR.(uni2.GE.1.D0).
     *             OR.(uni3.GE.1.D0).OR.(uni4.GE.1.d0).
     *             OR.(uni5.GE.1.D0)) then
                 uni1=drand(0)
                 uni2=drand(0)
                 uni3=drand(0)
                 uni4=drand(0)
                 uni5=drand(0)
                 goto 113
                 endif
                 auxgumb1=exp(log(uni5)/r)/
     *                  (exp(log(uni5)/r) + exp(log(1-uni5)/r))
                 if(uni4.le.1/r) then
                                 auxgumb2=uni1*uni2
                                 else
                                 auxgumb2=uni3
                 endif
                 ugumbel=exp(auxgumb1*log(auxgumb2)/AfoncGB(auxgumb1))
                 vgumbel=exp((1-auxgumb1)*log(auxgumb2)/AfoncGB(auxgumb1))
                 uindep=0.D0
                 vindep=0.D0
 114             if (uindep.LE.0.D0.OR.vindep.LE.0.D0.
     *              OR.uindep.GE.1.D0.OR.vindep.GE.1.D0) then
                 uindep=drand(0)
                 vindep=drand(0)
                 goto 114
                 endif
                 u(i)=max(uindep**(1/(1-al)),ugumbel**(1/al))
                 v(i)=max(vindep**(1/(1-be)),vgumbel**(1/be))
111           continue
        end
****************************************************************************
PROCEDURE GENERANT 100 COUPLES (u,v) DE COPULE ASYMETRIQUE DE TAWN AVEC
 alpha=0.78, beta=0.97 et r=50
****************************************************************************
      subroutine gumasym6(depart,u,v)
      integer n, d, m, k, kasym
      parameter (n=100, d=100, m=500, k=1, kasym=3)
      integer i
      real*8 depart,
     *        uni1,uni2,uni3,uni4,uni5,auxgumb1,auxgumb2,r,
     *        pt, drand, rand,ugumbel,vgumbel,uindep,vindep,
     *        AfoncGB, AfoncGBasym,al,be
      real*8 u(n), v(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external drand,rand
      external lgvraisasym, lgvrais
      external phi, D01AMF
      external AfoncGB, AfoncGBasym
       r=50.D0
       al=0.78d0
       be=0.97d0
            do 111 i=1,n
                 uni1=0.D0
                 uni2=0.D0
                 uni3=0.D0
                 uni4=0.D0
                 uni5=0.D0
 113             if((uni1.LE.0.D0).OR.(uni2.LE.0.D0).
     *             OR.(uni3.LE.0.D0).OR.(uni4.LE.0.d0).
     *             OR.(uni5.LE.0.D0).OR.(uni1.GE.1.D0).
     *             OR.(uni2.GE.1.D0).
     *             OR.(uni3.GE.1.D0).OR.(uni4.GE.1.d0).
     *             OR.(uni5.GE.1.D0)) then
                 uni1=drand(0)
                 uni2=drand(0)
                 uni3=drand(0)
                 uni4=drand(0)
                 uni5=drand(0)
                 goto 113
                 endif
                 auxgumb1=exp(log(uni5)/r)/
     *                  (exp(log(uni5)/r) + exp(log(1-uni5)/r))
                 if(uni4.le.1/r) then
                                 auxgumb2=uni1*uni2
                                 else
                                 auxgumb2=uni3
                 endif
                 ugumbel=exp(auxgumb1*log(auxgumb2)/AfoncGB(auxgumb1))
                 vgumbel=exp((1-auxgumb1)*log(auxgumb2)/AfoncGB(auxgumb1))
                 uindep=0.D0
                 vindep=0.D0
 114             if (uindep.LE.0.D0.OR.vindep.LE.0.D0.
     *              OR.uindep.GE.1.D0.OR.vindep.GE.1.D0) then
                 uindep=drand(0)
                 vindep=drand(0)
                 goto 114
                 endif
                 u(i)=max(uindep**(1/(1-al)),ugumbel**(1/al))
                 v(i)=max(vindep**(1/(1-be)),vgumbel**(1/be))
111           continue
        end

****************************************************************************
FONCTION RETOURNANT LA VALEUR DE LA FONCTION DE DEPENDANCE D'UNE COPULE DE TYPE
 GUMBEL B. DE PARAMETRE r>=1
****************************************************************************
      real*8 function AfoncGB(t)
      real*8 t,r
      common /var1/ r
      AfoncGB=((1-t)**r + t**r)**(1/r)
      end
****************************************************************************
FONCTION RETOURNANT LA VALEUR DE LA FONCTION DE DEPENDANCE D'UNE COPULE DE TYPE
 ASYMETRIQUE DE TAWN. DE PARAMETRES r>=1, et 0 <= a, b <= 1
****************************************************************************

      real*8 function AfoncGBasym(t)
      real*8 t,r, al,be
      common /var1/ r
      common /var2/ al,be
      AfoncGBasym=1-be+(be-al)*t+ ((be*(1-t))**r + (al*t)**r)**(1/r)
      end
****************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE JOE ET AL. AVEC UN SEUIL DE 5/50 POUR UN
 ECHANTILLON DE TAILLE 50
****************************************************************************
       subroutine joe505(xvec,yvec,taut7,Astarn)
       include 'commun50'
       integer rgx(n), rgy(n), antirgx(n), antirgy(n),antirgt(n),
     *        ind,astar,i,j
       real*8 xvec(n),yvec(n), xord(n), yord(n),pas,tord(n)
       real*8 Astarn(d),t(d),S,e(n)
```

```
        real*8 sum

        real*8 taut7,taus,integ(d)
        real*8 Q,z1(n),z2(n)
        do 199 ind=1,d
               t(ind)=dble(ind)/dble(d)
199            continue
        pas=t(2)-t(1)
        nstar=20.d0
        call indexx(n,xvec,antirgx)
        call indexx(n,yvec,antirgy)
        do 301 i=1,n
               xord(i)=xvec(antirgx(i))
               yord(i)=yvec(antirgy(i))
301            continue
       call rank(n,antirgx,rgx)
       call rank(n,antirgy,rgy)
       do 302 i=1,n
               z1(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
               z2(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
302     continue
        do 305 ind=1,d-1
               B=(1-t(ind))/t(ind)

        do 306 i=1,n
        Q=B*z2(i)
               if (z1(i).GT.Q) then
                      e(i)=z1(i)/n
               else
                      e(i)=Q/n
               endif
306     enddo
        call indexx(n,e,antirgt)
        do 303 i=1,n
               tord(i)=e(antirgt(i))
303     continue
        sum=0.d0
        j=20.d0
        do 304 i=31,45
        sum=tord(i)*j+sum
        j=j-1.d0
304     enddo
        Astarn(ind)=t(ind)*sum/15.d0
305     enddo
        call tauK(t,pas,Astarn,taut7,taus,integ)
        end
********************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE JOE ET AL. AVEC UN SEUIL DE 10/50 POUR UN
 ECHANTILLON DE TAILLE 50
********************************************************************************
        subroutine joe5010(xvec,yvec,taut7,Astarn)
        include 'commun50'
        integer rgx(n), rgy(n), antirgx(n), antirgy(n),antirgt(n),
     *       ind,nstar,i,j
        real*8 xvec(n),yvec(n), xord(n), yord(n),pas,tord(n)
        real*8 Astarn(d),t(d),B,e(n)
        real*8 sum

        real*8 taut7,taus,integ(d)
        real*8 Q,z1(n),z2(n)
        do 199 ind=1,d
               t(ind)=dble(ind)/dble(d)
199            continue
        pas=t(2)-t(1)
        nstar=10.d0
        call indexx(n,xvec,antirgx)
        call indexx(n,yvec,antirgy)
        do 301 i=1,n
               xord(i)=xvec(antirgx(i))
               yord(i)=yvec(antirgy(i))
301            continue
       call rank(n,antirgx,rgx)
       call rank(n,antirgy,rgy)
       do 302 i=1,n
               z1(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
               z2(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
302     continue
        do 305 ind=1,d-1
               B=(1-t(ind))/t(ind)
        do 306 i=1,n
        Q=B*z2(i)
               if (z1(i).GT.Q) then
                      e(i)=z1(i)/n
               else
                      e(i)=Q/n
               endif
306     enddo
        call indexx(n,e,antirgt)
        do 303 i=1,n
               tord(i)=e(antirgt(i))
303     continue
        sum=0.d0
        j=10.d0
        do 304 i=41,45
        sum=tord(i)*j+sum
        j=j-1.d0
304     enddo
        Astarn(ind)=t(ind)*sum/5.d0
305     enddo
        call tauK(t,pas,Astarn,taut7,taus,integ)
        end
********************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE JOE ET AL. AVEC UN SEUIL DE 10/100 POUR UN
 ECHANTILLON DE TAILLE 100 ******************************************************
***************************
        subroutine joe10010(xvec,yvec,taut7,Astarn)
        include 'commun100'
        integer rgx(n), rgy(n), antirgx(n), antirgy(n),antirgt(n),
     *       ind,nstar,i,j
        real*8 xvec(n),yvec(n), xord(n), yord(n),pas,tord(n)
        real*8 Astarn(d),t(d),B,e(n)
        real*8 sum

        real*8 taut7,taus,integ(d)
        real*8 Q,z1(n),z2(n)
        do 199 ind=1,d
               t(ind)=dble(ind)/dble(d)
199            continue
        pas=t(2)-t(1)
        nstar=20.d0
        call indexx(n,xvec,antirgx)
        call indexx(n,yvec,antirgy)
        do 301 i=1,n
               xord(i)=xvec(antirgx(i))
               yord(i)=yvec(antirgy(i))
301            continue
       call rank(n,antirgx,rgx)
       call rank(n,antirgy,rgy)
       do 302 i=1,n
               z1(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
               z2(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
302     continue
        do 305 ind=1,d-1
               B=(1-t(ind))/t(ind)
        do 306 i=1,n
        Q=B*z2(i)
               if (z1(i).GT.Q) then
                      e(i)=z1(i)/n
               else
                      e(i)=Q/n
               endif
306     enddo
        call indexx(n,e,antirgt)
        do 303 i=1,n
               tord(i)=e(antirgt(i))
303     continue
        sum=0.d0
        j=20.d0
        do 304 i=81,95
        sum=tord(i)*j+sum
        j=j-1.d0
304     enddo
        Astarn(ind)=t(ind)*sum/15.d0
305     enddo
        call tauK(t,pas,Astarn,taut7,taus,integ)
        end
********************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE JOE ET AL. AVEC UN SEUIL DE 10/100 POUR UN
 ECHANTILLON DE TAILLE 100
********************************************************************************
        subroutine joe10020(xvec,yvec,taut7,Astarn)
        include 'commun100'
        integer rgx(n), rgy(n), antirgx(n), antirgy(n),antirgt(n),
     *       ind,nstar,i,j

        real*8 xvec(n),yvec(n), xord(n), yord(n),pas,tord(n)
        real*8 Astarn(d),t(d),B,e(n)
        real*8 sum

        real*8 taut7,taus,integ(d)
        real*8 Q,z1(n),z2(n)
        do 199 ind=1,d
               t(ind)=dble(ind)/dble(d)
199            continue
        pas=t(2)-t(1)
        nstar=10.d0
        call indexx(n,xvec,antirgx)
        call indexx(n,yvec,antirgy)
        do 301 i=1,n
               xord(i)=xvec(antirgx(i))
               yord(i)=yvec(antirgy(i))
301            continue
       call rank(n,antirgx,rgx)
       call rank(n,antirgy,rgy)
       do 302 i=1,n
               z1(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
               z2(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
302     continue
        do 305 ind=1,d-1
               B=(1-t(ind))/t(ind)
        do 306 i=1,n
        Q=B*z2(i)
               if (z1(i).GT.Q) then
                      e(i)=z1(i)/n
               else
                      e(i)=Q/n
               endif
306     enddo
        call indexx(n,e,antirgt)
        do 303 i=1,n
               tord(i)=e(antirgt(i))
303     continue
        sum=0.d0
        j=10.d0
        do 304 i=91,95
        sum=tord(i)*j+sum
        j=j-1.d0
304     enddo
        Astarn(ind)=t(ind)*sum/5.d0
305     enddo
        call tauK(t,pas,Astarn,taut7,taus,integ)
        end
********************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 5/50 POUR UN
 ECHANTILLON DE TAILLE 50
********************************************************************************
        subroutine kho505(xvec,yvec,taut1,Astarn)
        include 'commun50'
       integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *       istar(n),k,ind,nstar,i
        real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     -       xstar(n), ystar(n), rvec(n), wvec(n),rstar21
        real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)

        real*8 taut1,taus,integ(d)
        do 199 ind=1,d
               t(ind)=dble(ind)/dble(d)
199            continue
        pas=t(2)-t(1)
        rstar21=5.d0
        nstar=0.d0
        call indexx(n,xvec,antirgx)
        call indexx(n,yvec,antirgy)
        do 101 i=1,n
               xord(i)=xvec(antirgx(i))
               yord(i)=yvec(antirgy(i))
101            continue
        call rank(n,antirgx,rgx)
        call rank(n,antirgy,rgy)
        do i=1,n
                rvec(i)=0.d0
        enddo
        do 102 i=1,n
               xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
               ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
               rvec(i)=xstar(i)+ystar(i)
               if (rvec(i).gt.rstar21) then
                        nstar=nstar+1
                        istar(nstar)=i
               endif
102            continue
       do 103 k=1,nstar
               wvec(k)=xstar(istar(k))/rvec(istar(k))
103            continue
       do 104 ind=1,d-1
               Anaux=0.d0
               do 105 k=1,nstar
                       auxil1=wvec(k)*(t(ind))
                       auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
                       if (auxil1.ge.auxil2) then
                           Anaux=Anaux+auxil1
                       else
                           Anaux=Anaux+auxil2
                       endif
105                    continue
               Astarn(ind)=2.d0*Anaux/dble(nstar)
104            continue
        call tauK(t,pas,Astarn,taut1,taus,integ)
        end
********************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 10/50 POUR UN
 ECHANTILLON DE TAILLE 50
********************************************************************************
        subroutine kho5010(xvec,yvec,taut1,Astarn)
        include 'commun50'
        integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *       istar(n),k,ind,nstar,i
        real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     *       xstar(n), ystar(n), rvec(n), wvec(n),rstar21
        real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)

        real*8 taut1,taus,integ(d)
        do 199 ind=1,d
               t(ind)=dble(ind)/dble(d)
199            continue
        pas=t(2)-t(1)
        rstar21=10.d0
        nstar=0.d0
        call indexx(n,xvec,antirgx)
        call indexx(n,yvec,antirgy)
        do 101 i=1,n
               xord(i)=xvec(antirgx(i))
               yord(i)=yvec(antirgy(i))
101            continue
        call rank(n,antirgx,rgx)
        call rank(n,antirgy,rgy)
        do i=1,n
                rvec(i)=0.d0
        enddo
        do 102 i=1,n
```

```
          xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
          ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
          rvec(i)=xstar(i)+ystar(i)
          if (rvec(i).gt.rstar21) then
                              nstar=nstar+1
                              istar(nstar)=i
                          endif
102       continue
      do 103 k=1,nstar
          wvec(k)=xstar(istar(k))/rvec(istar(k))
103       continue
      do 104 ind=1,d-1
          Anaux=0.d0
          do 105 k=1,nstar
              auxil1=wvec(k)*(t(ind))
              auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
              if (auxil1.ge.auxil2) then
                  Anaux=Anaux+auxil1
              else
                  Anaux=Anaux+auxil2
              endif
105           continue
          Astarn(ind)=2.d0*Anaux/dble(nstar)
104       continue
       call tauK(t,pas,Astarn,taut1,taus,integ)
       end
****************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 10/100 POUR UN
 ECHANTILLON DE TAILLE 100
****************************************************************************
       subroutine kho10010(xvec,yvec,taut1,Astarn)
       include 'commun100'
       integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *       istar(n),k,ind,nstar,i
       real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     *       xstar(n), ystar(n), rvec(n), wvec(n),rstar21
       real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)

       real*8 taut1,taus,integ(d)
       do 199 ind=1,d
           t(ind)=dble(ind)/dble(d)
199        continue
       pas=t(2)-t(1)
       rstar21=10.d0
       nstar=0.d0
       call indexx(n,xvec,antirgx)
       call indexx(n,yvec,antirgy)
       do 101 i=1,n
           xord(i)=xvec(antirgx(i))
           yord(i)=yvec(antirgy(i))
101        continue
       call rank(n,antirgx,rgx)
       call rank(n,antirgy,rgy)
       do i=1,n
              rvec(i)=0.d0
       enddo
       do 102 i=1,n
          xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
          ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
          rvec(i)=xstar(i)+ystar(i)
          if (rvec(i).gt.rstar21) then
                              nstar=nstar+1
                              istar(nstar)=i
                          endif
102       continue
      do 103 k=1,nstar
          wvec(k)=xstar(istar(k))/rvec(istar(k))
103       continue
      do 104 ind=1,d-1
          Anaux=0.d0
          do 105 k=1,nstar
              auxil1=wvec(k)*(t(ind))
              auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
              if (auxil1.ge.auxil2) then
                  Anaux=Anaux+auxil1
              else
                  Anaux=Anaux+auxil2
              endif
105           continue
          Astarn(ind)=2.d0*Anaux/dble(nstar)
104       continue
       call tauK(t,pas,Astarn,taut1,taus,integ)
       end
****************************************************************************
PROCEDURE CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 20/100 POUR UN
 ECHANTILLON DE TAILLE 100
****************************************************************************
       subroutine kho10020(xvec,yvec,taut1,Astarn)
       include 'commun100'
       integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *       istar(n),k,ind,nstar,i
       real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     *       xstar(n), ystar(n), rvec(n), wvec(n),rstar21
       real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)

       real*8 taut1,taus,integ(d)
       do 199 ind=1,d
           t(ind)=dble(ind)/dble(d)
199        continue
       pas=t(2)-t(1)
       rstar21=20.d0
       nstar=0.d0
       call indexx(n,xvec,antirgx)
       call indexx(n,yvec,antirgy)
       do 101 i=1,n
           xord(i)=xvec(antirgx(i))
           yord(i)=yvec(antirgy(i))
101        continue
       call rank(n,antirgx,rgx)
       call rank(n,antirgy,rgy)
       do i=1,n
              rvec(i)=0.d0
       enddo
       do 102 i=1,n
          xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
          ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
          rvec(i)=xstar(i)+ystar(i)
          if (rvec(i).gt.rstar21) then
                              nstar=nstar+1
                              istar(nstar)=i
                          endif
102       continue
      do 103 k=1,nstar
          wvec(k)=xstar(istar(k))/rvec(istar(k))
103       continue
      do 104 ind=1,d-1
          Anaux=0.d0
          do 105 k=1,nstar
              auxil1=wvec(k)*(t(ind))
              auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
              if (auxil1.ge.auxil2) then
                  Anaux=Anaux+auxil1
              else
                  Anaux=Anaux+auxil2
              endif
105           continue
          Astarn(ind)=2.d0*Anaux/dble(nstar)
104       continue
       call tauK(t,pas,Astarn,taut1,taus,integ)
       end
****************************************************************************
PROCEDURE DE CALCUL DE L'ESTIMATEUR DE CAPERAA ET AL. POUR UN ECHANTILLON DE
 TAILLE n=50
****************************************************************************
      subroutine CFG1(u,v,taut,nous1)
      integer n, d
      parameter (n=50, d=100)
      integer i, ind, indi, k2, antirgz(n)
      real*8 somv, somu, aux, pas, taut, taus
      real*8 x(d), auxAcfg(d), Acfg(d), integ(d), p1(d),
     *        logA0(d), logA1(d)
      real*8 u(n), v(n), z(n), zord(n)
       do 199 indi=1,d
            x(indi)=dble(indi)/dble(d)
199    continue
       pas=x(2)-x(1)
      do 10 indi=1,d
           p1(indi)=1-x(indi)
10    continue
      somu=0.0D0
      somv=0.0D0
      do 99 indi=1,d
           auxAcfg(indi)=1.0D0
           Acfg(indi)=0.0D0
 99      continue
      do 12 i=1,n
           z(i)=log(u(i))/(log(u(i)*v(i)))
12    continue
      call indexx(n,z,antirgz)
      do 77 i=1,n
         zord(i)=z(antirgz(i))
77    continue
        do ind=1,d
                logA0(ind)=0.d0
                logA1(ind)=0.d0
                Acfg(ind)=0.d0
        enddo
      ind=1
700   if (x(ind).le.z(antirgz(1))) then
                logA0(ind)= log(1-x(ind))
                do 37 i=1,n
                    logA1(ind)=logA1(ind)+log(z(antirgz(i)))-
     *                          log(1-z(antirgz(i)))
37              continue
                logA1(ind)=logA1(ind)/dble(n) + log(1-x(ind))
                nous1(ind) = exp(p1(ind)*logA0(ind)+
     *                         (1-p1(ind))*logA1(ind))
                nous2(ind) = exp(p2(ind)*logA0(ind)+
     *                         (1-p2(ind))*logA1(ind))
                ind=ind+1
      goto 700
      endif
      k2=1
      aux=z(antirgz(n))
800   if (x(ind).le.aux) then
801       if (x(ind).le.z(antirgz(k2+1))) then
              do 26 i=1,k2
                  logA0(ind)=logA0(ind)+log(1-z(antirgz(i)))
     *                         -log(z(antirgz(i)))
26            continue
              logA0(ind)=logA0(ind)/dble(n)
     *                   + dble(k2)*log(x(ind))/dble(n)
     *                   + (1-dble(k2)/dble(n))*log(1-x(ind))
              do 27 i=k2+1,n
                  logA1(ind)=logA1(ind)+log(z(antirgz(i)))
     *                         -log(1-z(antirgz(i)))
27            continue
              logA1(ind)=logA1(ind)/dble(n)
     *                   +dble(k2)*log(x(ind))/dble(n)
     *                   +(1-dble(k2)/dble(n))*log(1-x(ind))
              nous1(ind)=exp(p1(ind)*logA0(ind)
     *                   + (1-p1(ind))*logA1(ind))
              nous2(ind)=exp(p2(ind)*logA0(ind)
     *                   + (1-p2(ind))*logA1(ind))
              ind=ind+1
              goto 801
          endif
          k2=k2+1
          goto 800
      endif
900   if (ind.le.d) then
         do 28 i=1,n
              logA0(ind)=logA0(ind)+log(1-z(antirgz(i)))-
     *                      log(z(antirgz(i)))
28       continue
          logA0(ind)=logA0(ind)/dble(n) - log(x(ind))
          logA1(ind)=log(x(ind))
          nous1(ind)=exp(p1(ind)*logA0(ind)+(1-p1(ind))*logA1(ind))
          nous2(ind)=exp(p2(ind)*logA0(ind)+(1-p2(ind))*logA1(ind))
          ind=ind+1
         goto 900
      endif
        call tauf(x,pas,nous1,taut,taus,integ)
        end
****************************************************************************
PROCEDURE DE CALCUL DE L'ESTIMATEUR DE CAPERAA ET AL. POUR UN ECHANTILLON DE
 TAILLE n=50
****************************************************************************
      subroutine CFG2(u,v,taut,nous1)
      integer n, d
      parameter (n=100, d=100)
      integer i, ind, indi, k2, antirgz(n)
      real*8 somv, somu, aux, pas, taut, taus
      real*8 x(d), auxAcfg(d), Acfg(d), integ(d), p1(d),
     *        logA0(d), logA1(d)
      real*8 u(n), v(n), z(n), zord(n)
       do 199 indi=1,d
            x(indi)=dble(indi)/dble(d)
199    continue
       pas=x(2)-x(1)
      do 10 indi=1,d
           p1(indi)=1-x(indi)
10    continue
      somu=0.0D0
      somv=0.0D0
      do 99 indi=1,d
           auxAcfg(indi)=1.0D0
           Acfg(indi)=0.0D0
 99      continue
      do 12 i=1,n
           z(i)=log(u(i))/(log(u(i)*v(i)))
12    continue
      call indexx(n,z,antirgz)
      do 77 i=1,n
         zord(i)=z(antirgz(i))
77    continue
        do ind=1,d
                logA0(ind)=0.d0
                logA1(ind)=0.d0
                Acfg(ind)=0.d0
        enddo
      ind=1
700   if (x(ind).le.z(antirgz(1))) then
                logA0(ind)= log(1-x(ind))
                do 37 i=1,n
                    logA1(ind)=logA1(ind)+log(z(antirgz(i)))-
     *                          log(1-z(antirgz(i)))
37              continue
                logA1(ind)=logA1(ind)/dble(n) + log(1-x(ind))
                nous1(ind) = exp(p1(ind)*logA0(ind)+
     *                         (1-p1(ind))*logA1(ind))
                nous2(ind) = exp(p2(ind)*logA0(ind)+
     *                         (1-p2(ind))*logA1(ind))
                ind=ind+1
      goto 700
      endif
      k2=1
      aux=z(antirgz(n))
800   if (x(ind).le.aux) then
801       if (x(ind).le.z(antirgz(k2+1))) then
              do 26 i=1,k2
                  logA0(ind)=logA0(ind)+log(1-z(antirgz(i)))
     *                         -log(z(antirgz(i)))
26            continue
              logA0(ind)=logA0(ind)/dble(n)
     *                   + dble(k2)*log(x(ind))/dble(n)
     *                   + (1-dble(k2)/dble(n))*log(1-x(ind))
              do 27 i=k2+1,n
                  logA1(ind)=logA1(ind)+log(z(antirgz(i)))
     *                         -log(1-z(antirgz(i)))
```

```
27            continue
              logA1(ind)=logA1(ind)/dble(n)
     *                    -dble(k2)*log(x(ind))/dble(n)
     *                    +(1-dble(k2)/dble(n))*log(1-x(ind))
              nous1(ind)=exp(p1(ind)*logA0(ind)
     *                    + (1-p1(ind))*logA1(ind))
              nous2(ind)=exp(p2(ind)*logA0(ind)
     *                    + (1-p2(ind))*logA1(ind))
              ind=ind+1
              goto 801
          endif
          k2=k2+1
          goto 800
        endif
900     if (ind.le.d) then
          do 28 i=1,n
               logA0(ind)=logA0(ind)+log(1-x(antirgx(i)))-
     *                    log(x(antirgx(i)))
28        continue
           logA0(ind)=logA0(ind)/dble(n) + log(x(ind))
           logA1(ind)=log(x(ind))
           nous1(ind)=exp(p1(ind)*logA0(ind)+(1-p1(ind))*logA1(ind))
           nous2(ind)=exp(p2(ind)*logA0(ind)+(1-p2(ind))*logA1(ind))
           ind=ind+1
          goto 900
        endif
         call tauK(x,pas,nous1,taut,taus,integ)
         end
```

```
*****************************************************************************
PROCEDURE CALCULANT L'ERREUR L1 ENTRE LA FONCTION A DE TYPE GUMBEL B AVEC r=4/3
 ET  L'ESTIMATION Astarnl.
*****************************************************************************
       subroutine errlGS1(Astarnl,erreurL)
       include 'commun'
       integer ind,q
       real*8 ra
       parameter(q=100,ra=1.333333d0)
       integer l
       real*8 Astarnbar(d),Astarnl(d,d),erreurL1(d),j,erreurL(d)
       j=0.01d0
       do ind=1,d-1
              Astarnbar(ind)=((j**ra)+((1-j)**ra))**(1/ra)
              j=j+0.01d0
       enddo
       do l=1,q
       erreurL1(l)=0.d0
       enddo
       do l=1,q
       do ind=1,d-1
              erreurL1(l)=erreurL1(l)+abs(Astarnbar(ind)-Astarnl(ind,l))
       enddo
       enddo
       do l=1,q
       erreurL(l)=erreurL1(l)*0.01d0
       enddo
       end
*****************************************************************************
PROCEDURE CALCULANT L'ERREUR L1 ENTRE LA FONCTION A DE TYPE GUMBEL B AVEC r=2
 ET  L'ESTIMATION Astarnl.
*****************************************************************************
       subroutine errlGS2(Astarnl,erreurL)
       include 'commun'
       integer ind,q
       real*8 ra
       parameter(q=100,ra=2.d0)
       integer l
       real*8 Astarnbar(d),Astarnl(d,d),erreurL1(d),j,erreurL(d)
       j=0.01d0
       do ind=1,d-1
              Astarnbar(ind)=((j**ra)+((1-j)**ra))**(1/ra)
              j=j+0.01d0
       enddo
       do l=1,q
       erreurL1(l)=0.d0
       enddo
       do l=1,q
       do ind=1,d-1
              erreurL1(l)=erreurL1(l)+abs(Astarnbar(ind)-Astarnl(ind,l))

       enddo
       enddo
       do l=1,q
       erreurL(l)=erreurL1(l)*0.01d0
       enddo
       end
*****************************************************************************
PROCEDURE CALCULANT L'ERREUR L1 ENTRE LA FONCTION A DE TYPE GUMBEL B AVEC r=4
 ET  L'ESTIMATION Astarnl.
*****************************************************************************
       subroutine errlGS1(Astarnl,erreurL)
       include 'commun'
       integer ind,q
       real*8 ra
       parameter(q=100,ra=4.d0)
       integer l
       real*8 Astarnbar(d),Astarnl(d,d),erreurL1(d),j,erreurL(d)
       j=0.01d0
       do ind=1,d-1
              Astarnbar(ind)=((j**ra)+((1-j)**ra))**(1/ra)
              j=j+0.01d0
       enddo
       do l=1,q
       erreurL1(l)=0.d0
       enddo
       do l=1,q
       do ind=1,d-1
              erreurL1(l)=erreurL1(l)+abs(Astarnbar(ind)-Astarnl(ind,l))

       enddo
       enddo
       do l=1,q
       erreurL(l)=erreurL1(l)*0.01d0
       enddo
       end
*****************************************************************************
PROCEDURE CALCULANT L'ERREUR L1 ENTRE LA FONCTION A ASYMETRIQUE DE TAWN AVEC
 alpha=0.78, beta=0.97 et r=1.42   ET  L'ESTIMATION Astarnl.
*****************************************************************************
       subroutine errlGAS1(Astarnl,erreurL)
       include 'commun'
       integer ind,q
       real*8 ra,a,b
       parameter(q=100,a=0.78d0,b=0.97d0,ra=1.42d0)
       integer l
       real*8 Astarnbar(d),Astarnl(d,d),erreurL1(d),j,erreurL(d)
       j=0.01d0
       do ind=1,d-1
       Astarnbar(ind)=1-b+(b-a)*j+((a**r)*(j**r)+(b**r)*((1-j)**r))**(1/r)
              j=j+0.01d0
       enddo
       do l=1,q
       erreurL1(l)=0.d0
       enddo
       do l=1,q
       do ind=1,d-1
              erreurL1(l)=erreurL1(l)+abs(Astarnbar(ind)-Astarnl(ind,l))

       enddo
       enddo
       do l=1,q
       erreurL(l)=erreurL1(l)*0.01d0
       enddo
       end
*****************************************************************************
PROCEDURE CALCULANT L'ERREUR L1 ENTRE LA FONCTION A ASYMETRIQUE DE TAWN AVEC
 alpha=0.78, beta=0.97 et r=2.58   ET  L'ESTIMATION Astarnl.
*****************************************************************************
       subroutine errlGAS2(Astarnl,erreurL)
       include 'commun'
       integer ind,q
       real*8 ra,a,b
       parameter(q=100,a=0.78d0,b=0.97d0,ra=2.58d0)
       integer l
       real*8 Astarnbar(d),Astarnl(d,d),erreurL1(d),j,erreurL(d)
       j=0.01d0
       do ind=1,d-1


       Astarnbar(ind)=1-b+(b-a)*j+((a**r)*(j**r)+(b**r)*((1-j)**r))**(1/r)
              j=j+0.01d0
       enddo
       do l=1,q
       erreurL1(l)=0.d0
       enddo
       do l=1,q
       do ind=1,d-1
              erreurL1(l)=erreurL1(l)+abs(Astarnbar(ind)-Astarnl(ind,l))
       enddo
       enddo
       do l=1,q
       erreurL(l)=erreurL1(l)*0.01d0
       enddo
       end
*****************************************************************************
PROCEDURE CALCULANT L'ERREUR L1 ENTRE LA FONCTION A ASYMETRIQUE DE TAWN AVEC
 alpha=0.78, beta=0.97 et r=50   ET  L'ESTIMATION Astarnl.
*****************************************************************************
       subroutine errlGAS3(Astarnl,erreurL)
       include 'commun'
       integer ind,q
       real*8 ra,a,b
       parameter(q=100,a=0.78d0,b=0.97d0,ra=50.d0)
       integer l
       real*8 Astarnbar(d),Astarnl(d,d),erreurL1(d),j,erreurL(d)
       j=0.01d0
       do ind=1,d-1
       Astarnbar(ind)=1-b+(b-a)*j+((a**r)*(j**r)+(b**r)*((1-j)**r))**(1/r)
              j=j+0.01d0
       enddo
       do l=1,q
       erreurL1(l)=0.d0
       enddo
       do l=1,q
       do ind=1,d-1
              erreurL1(l)=erreurL1(l)+abs(Astarnbar(ind)-Astarnl(ind,l))
       enddo
       enddo
       do l=1,q
       erreurL(l)=erreurL1(l)*0.01d0
       enddo
       end
*****************************************************************************
FICHIER DE DECLARATIONS DE VARIABLES COMMUNES AUX DIFFERENTS PROGRAMMES ET
 SOUS-PROGRAMMES (commun50)
*****************************************************************************
      implicit none
      integer n, d, nrepli
      parameter (n=50, d=100, nrepli=50)
      real*8 r, alpha, m, parA, c, ord
      real*8 drand
      common /var1/ r
      common /var2/ m, alpha, c, ord
      common /var3/ parA
      external drand
*****************************************************************************
FICHIER DE DECLARATIONS DE VARIABLES COMMUNES AUX DIFFERENTS PROGRAMMES ET
 SOUS-PROGRAMMES (commun100)
*****************************************************************************
      implicit none
      integer n, d, nrepli
      parameter (n=100, d=100, nrepli=50)
      real*8 r, alpha, m, parA, c, ord
      real*8 drand
      common /var1/ r
      common /var2/ m, alpha, c, ord
      common /var3/ parA
      external drand
*****************************************************************************
POUR LES PROCEDURE indexx, tauK, rank, VOIR ANNEXE E
*****************************************************************************
```

Annexe H

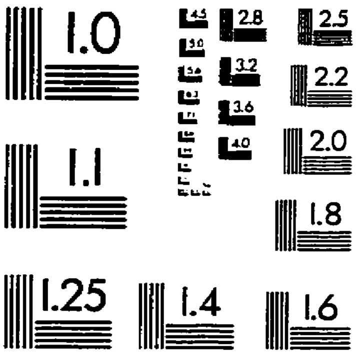
Programmes informatiques de fortran 77 utilisés pour l'analyse de données

```
*******************************************************************************
PROGRAMME PRINCIPAL CALCULANT L'ESTIMATEUR DE CAPERAA ET AL. SUR L'ECHANTILLON
        DE DONNEES CONSTITUE DES MAXIMA DE L'ALCAN
*******************************************************************************
        program valext
        integer n,nn,i,j
        parameter (nn=52,n=42)
        integer rgx(n),rgy(n),antirgx(n),antirgy(n)
        real*8 x1(nn),x2(nn),x3(nn),x4(nn),x5(nn),u1(nn),v1(nn),x8(nn)
        real*8 taut,x(n),y(n)
        real*8 transfx(n),transfy(n)
        open(1,file='fort.4')
        do 10 i=1,nn
             read(1,*) x1(i),x2(i),x3(i),x4(i),x5(i),u1(i),v1(i),x8(i)
 10     continue
        j=1
        do i=11,nn
                x(j)=u1(i)
                y(j)=v1(i)
                j=j+1
        enddo
        do i=1,n
        enddo
        call indexx(n,x,antirgx)

        call indexx(n,y,antirgy)
        do i=1,n
        enddo
        call rank(n,antirgx,rgx)
        call rank(n,antirgy,rgy)
        do 11 i=1,n
          transfx(i)=dble(rgx(i))/dble(n+1)
          transfy(i)=dble(rgy(i))/dble(n+1)
 11     continue
        call CPG3(transfx,transfy,taut,nousl)
        print*,taut
        close(1)
        end
*******************************************************************************
  SOUS-PROGRAMME CALCULANT L'ESTIMATEUR DE CAPERAA ET AL.
*******************************************************************************
      subroutine CPG3(u,v,taut,nousl)
      integer n, d,m , k, kasym
      parameter (n=42, m=500, d=100, k=1, kasym=3)
      integer i, ind, indi, j, k2, antirgz(n)
      real*8 somv, somu, aux,
     *        pas,r,al,be,pt,srand, rand, taut
      real*8 x(d), nousl(d), pl(d), logA0(d), logA1(d)
      real*8 bnousl(m)
      real*8  u(n),v(n)
      real*8 z(n), zord(n)
      common /var1/ r
      common /var2/ al,be
      common /subdi/pt
      external srand, rand
      do 199 indi=1,d
             x(indi)=dble(indi)/dble(d)
199   continue
      pas=x(2)-x(1)
      do 10 indi=1,d
             pl(indi)=1-x(indi)
 10   continue
      somu=0.0D0
      somv=0.0D0
      do 55 j=1,m
           bnousl(j)=0.0d0
55    continue
      do 99 indi=1,d
             nousl(indi)=0.0D0
99     continue
      do 12 i=1,n
           z(i)=log(u(i))/(log(u(i)*v(i)))
12    continue
      call indexx(n,z,antirgz)
      do 77 i=1,n
         zord(i)=z(antirgz(i))
77    continue
      do ind=1,d
                logA0(ind)=0.d0
                logA1(ind)=0.d0
      enddo
      ind=1
700   if (x(ind).le.z(antirgz(1))) then
                logA0(ind)= log(1-x(ind))
                do 37 i=1,n
                   logA1(ind)=logA1(ind)+log(z(antirgz(i)))-
     *                        log(1-z(antirgz(i)))
37              continue
                logA1(ind)=logA1(ind)/dble(n) - log(1-x(ind))
                nousl(ind) = exp(pl(ind)*logA0(ind)+
     *                      (1-pl(ind))*logA1(ind))
                ind=ind+1
      goto 700
      endif
      k2=1
      aux=z(antirgz(n))
800   if (x(ind).le.aux) then
801       if (x(ind).le.z(antirgz(k2+1))) then
             do 26 i=1,k2
                  logA0(ind)=logA0(ind)+log(1-z(antirgz(i)))
     *                       -log(z(antirgz(i)))
26           continue
             logA0(ind)=logA0(ind)/dble(n)
     *                  + dble(k2)*log(x(ind))/dble(n)
     *                  + (1-dble(k2)/dble(n))*log(1-x(ind))
             do 27 i=k2+1,n
                  logA1(ind)=logA1(ind)+log(z(antirgz(i)))
     *                       -log(1-z(antirgz(i)))
27           continue
             logA1(ind)=logA1(ind)/dble(n)
     *                  +dble(k2)*log(x(ind))/dble(n)
     *                  -(1-dble(k2)/dble(n))*log(1-x(ind))

             nousl(ind)=exp(pl(ind)*logA0(ind)
     *                  + (1-pl(ind))*logA1(ind))
             ind=ind+1
             goto 801
         endif
         k2=k2+1
         goto 800
      endif
900   if (ind.le.d) then
         do 28 i=1,n
              logA0(ind)=logA0(ind)+log(1-z(antirgz(i)))-
     *                   log(z(antirgz(i)))
28       continue
         logA0(ind)=logA0(ind)/dble(n) + log(x(ind))
         logA1(ind)=log(x(ind))
         nousl(ind)=exp(pl(ind)*logA0(ind)+(1-pl(ind))*logA1(ind))
         ind=ind+1
        goto 900
      endif
        call tauK(x,pas,nousl,taut,taus,integ)
        end
*******************************************************************************
PROGRAMME PRINCIPAL CALCULANT LES ESTIMATEURS DE JOE ET AL. ET DE KHOUDRAJI SUR
LES ECHANTILLONS DE DONNEES 1,2,3 ET 4 DE L'ALCAN
*******************************************************************************
        program valext1
        integer n,nn,i,j
        parameter (nn=3731,n=42)
        real*8 u1(nn),v1(nn),u2(nn),v2(nn),u3(nn),v3(nn),u4(nn),v4(nn)
        real*8 taut,datau(nn),datav(nn)
        parameter(r1=25)
        open(1,file='ANCC11.LSJ')
        open(2,file='ANCC11.CD')
        do 10 i=1,nn
             read(1,*) datau(i),u1(i)
 10       continue
        do 20 i=1,nn
             read(2,*) datav(i),v1(i)
 20       continue
        j=1
        do i=911,1820
                u2(j)=u1(i)
                v2(j)=v1(i)
                j=j+1
        enddo
        j=1
        do i=1821,2730
                u3(j)=u1(i)
                v3(j)=v1(i)
                j=j+1
        enddo
        j=1
        do i=2731,3731
                u4(j)=u1(i)
                v4(j)=v1(i)
                j=j+1
        enddo
        call joe125(u1,v1,taut)
        print*,taut
        call joe125(u2,v2,taut)
        print*,taut
        call joe125(u3,v3,taut)
        print*,taut
        call joe225(u4,v4,taut)
        print*,taut
        call kho125(u1,v1,taut)
        print*,taut
        call kho125(u2,v2,taut)
        print*,taut
        call kho125(u3,v3,taut)
        print*,taut
        call kho225(u4,v4,taut)
        print*,taut
        call kho1100(u1,v1,taut)
        print*,taut
        call kho1100(u2,v2,taut)
        print*,taut
        call kho1100(u3,v3,taut)
        print*,taut
        call kho2100(u4,v4,taut)
        print*,taut
        call kho1200(u1,v1,taut)
        print*,taut
        call kho1200(u2,v2,taut)
        print*,taut
        call kho1200(u3,v3,taut)
        print*,taut
        call kho2200(u4,v4,taut)
        print*,taut
        end
*******************************************************************************
  SOUS-PROGRAMME CALCULANT L'ESTIMATEUR DE JOE ET AL. AVEC UN SEUIL DE 25/910
*******************************************************************************

        subroutine joe125(xvec,yvec,taut7)
        include 'commun910'
        integer rgx(n), rgy(n), antirgx(n), antirgy(n),antirgt(n),
     *       ind,nstar,i,j
        real*8 xvec(n),yvec(n), xord(n), yord(n),pas,tord(n)
        real*8 Astarn(d),t(d),B,e(n)
        real*8 sum
        real*8 taut7,taus,integ(d)
        real*8 Q,z1(n),z2(n)
        do 199 ind=1,d
             t(ind)=dble(ind)/dble(d)
199     continue
        pas=t(2)-t(1)
        nstar=73.d0
        call indexx(n,xvec,antirgx)
        call indexx(n,yvec,antirgy)
        do 301 i=1,n
              xord(i)=xvec(antirgx(i))
              yord(i)=yvec(antirgy(i))
301           continue
        call rank(n,antirgx,rgx)
        call rank(n,antirgy,rgy)
        do 302 i=1,n
              z1(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
              z2(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
302     continue
        do 305 ind=1,d-1
              B=(1-t(ind))/t(ind)
        do 306 i=1,n
        Q=B*z2(i)
              if (z1(i).GT.Q) then
                    e(i)=z1(i)/n
              else
                    e(i)=Q/n
              endif
306     enddo
        call indexx(n,e,antirgt)
        do 303 i=1,n
              tord(i)=e(antirgt(i))
303     continue
        sum=0.d0
        j=73.d0
        do 304 i=838,900
        sum=tord(i)*j+sum
        j=j-1.d0
304     enddo
        Astarn(ind)=t(ind)*sum/63.d0
305     enddo
        call tauK(t,pas,Astarn,taut7,taus,integ)
        end
*******************************************************************************
  SOUS-PROGRAMME CALCULANT L'ESTIMATEUR DE JOE ET AL. AVEC UN SEUIL DE 25/1001
*******************************************************************************

        subroutine joe225(xvec,yvec,taut7)
        include 'commun1001'
        integer rgx(n), rgy(n), antirgx(n), antirgy(n),antirgt(n),
     *       ind,nstar,i,j
        real*8 xvec(n),yvec(n), xord(n), yord(n),pas,tord(n)
        real*8 Astarn(d),t(d),B,e(n)
        real*8 sum
        real*8 taut7,taus,integ(d)
        real*8 Q,z1(n),z2(n)
        do 199 ind=1,d
             t(ind)=dble(ind)/dble(d)
199     continue
        pas=t(2)-t(1)
        nstar=80.d0
        call indexx(n,xvec,antirgx)
        call indexx(n,yvec,antirgy)
        do 301 i=1,n
              xord(i)=xvec(antirgx(i))
              yord(i)=yvec(antirgy(i))
301           continue
        call rank(n,antirgx,rgx)
        call rank(n,antirgy,rgy)
        do 302 i=1,n
              z1(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
              z2(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
302     continue
        do 305 ind=1,d-1
              B=(1-t(ind))/t(ind)
        do 306 i=1,n
        Q=B*z2(i)
              if (z1(i).GT.Q) then
                    e(i)=z1(i)/n
              else
                    e(i)=Q/n
              endif
306     enddo
        call indexx(n,e,antirgt)
        do 303 i=1,n
              tord(i)=e(antirgt(i))
```

```
103         continue
      sum=0.d0
      j=60.d0
      do 304 i=922,981
      sum=tord(i)*j+sum
      j=j-1.d0
304   enddo
      Astarn(ind)=t(ind)*sum/60.d0
      print*,ind,Astarn(ind)
305   enddo
      call tauK(t,pas,Astarn,taut7,taus,integ)
      end
**********************************************************************
 SOUS-PROGRAMME CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 25/910
**********************************************************************
      subroutine kho125(xvec,yvec,taut3)
      include 'commun910'
      integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *       istar(n),k,ind,nstar,i
      real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     *       xstar(n), ystar(n), rvec(n), wvec(n),rstar21
      real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)


      real*8 taut3,taus,integ(d)
      do 199 ind=1,d
            t(ind)=dble(ind)/dble(d)
199         continue
      pas=t(2)-t(1)
      rstar21=25.d0
      nstar=0.d0
      call indexx(n,xvec,antirgx)
      call indexx(n,yvec,antirgy)
      do 101 i=1,n
            xord(i)=xvec(antirgx(i))
            yord(i)=yvec(antirgy(i))
101         continue
      call rank(n,antirgx,rgx)
      call rank(n,antirgy,rgy)
      do i=1,n
              rvec(i)=0.d0
      enddo
      do 102 i=1,n
            xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
            ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
            rvec(i)=xstar(i)+ystar(i)
            if (rvec(i).gt.rstar21) then
                              nstar=nstar+1
                              istar(nstar)=i
                          endif
102         continue
      do 103 k=1,nstar
            wvec(k)=xstar(istar(k))/rvec(istar(k))
103         continue
      do 104 ind=1,d-1
            Anaux=0.d0
            do 105 k=1,nstar
                  auxil1=wvec(k)*(t(ind))
                  auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
                  if (auxil1.ge.auxil2) then
                      Anaux=Anaux+auxil1
                  else
                      Anaux=Anaux+auxil2
                  endif
105               continue
            Astarn(ind)=2.d0*Anaux/dble(nstar)
104         continue
      call tauK(t,pas,Astarn,taut3,taus,integ)
      end
**********************************************************************
 SOUS-PROGRAMME CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 25/1001
**********************************************************************
      subroutine kho225(xvec,yvec,taut3)
      include 'commun1001'
      integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *       istar(n),k,ind,nstar,i
      real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     *       xstar(n), ystar(n), rvec(n), wvec(n),rstar21
      real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)


      real*8 taut3,taus,integ(d)
      do 199 ind=1,d
            t(ind)=dble(ind)/dble(d)
199         continue
      pas=t(2)-t(1)
      rstar21=25.d0
      nstar=0.d0
      call indexx(n,xvec,antirgx)
      call indexx(n,yvec,antirgy)
      do 101 i=1,n
            xord(i)=xvec(antirgx(i))
            yord(i)=yvec(antirgy(i))
101         continue
      call rank(n,antirgx,rgx)
      call rank(n,antirgy,rgy)
      do i=1,n
              rvec(i)=0.d0
      enddo
      do 102 i=1,n
            xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
            ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
            rvec(i)=xstar(i)+ystar(i)
            if (rvec(i).gt.rstar21) then
                              nstar=nstar+1
                              istar(nstar)=i
                          endif
102         continue
      do 103 k=1,nstar
            wvec(k)=xstar(istar(k))/rvec(istar(k))
103         continue
      do 104 ind=1,d-1
            Anaux=0.d0
            do 105 k=1,nstar
                  auxil1=wvec(k)*(t(ind))
                  auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
                  if (auxil1.ge.auxil2) then
                      Anaux=Anaux+auxil1
                  else
                      Anaux=Anaux+auxil2
                  endif
105               continue
            Astarn(ind)=2.d0*Anaux/dble(nstar)
104         continue
      call tauK(t,pas,Astarn,taut3,taus,integ)
      end
**********************************************************************
 SOUS-PROGRAMME CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 50/910
**********************************************************************
      subroutine kho1100(xvec,yvec,taut3)
      include 'commun910'
      integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *       istar(n),k,ind,nstar,i
      real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     *       xstar(n), ystar(n), rvec(n), wvec(n),rstar21
      real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)


      real*8 taut3,taus,integ(d)
      do 199 ind=1,d
            t(ind)=dble(ind)/dble(d)
199         continue
      pas=t(2)-t(1)
      rstar21=50.d0
      nstar=0.d0
      call indexx(n,xvec,antirgx)
      call indexx(n,yvec,antirgy)
      do 101 i=1,n
            xord(i)=xvec(antirgx(i))
            yord(i)=yvec(antirgy(i))
101         continue
      call rank(n,antirgx,rgx)
      call rank(n,antirgy,rgy)
      do i=1,n
              rvec(i)=0.d0
      enddo
      do 102 i=1,n
            xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
            ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
            rvec(i)=xstar(i)+ystar(i)
            if (rvec(i).gt.rstar21) then
                              nstar=nstar+1
                              istar(nstar)=i
                          endif
102         continue
      do 103 k=1,nstar
            wvec(k)=xstar(istar(k))/rvec(istar(k))
103         continue
      do 104 ind=1,d-1
            Anaux=0.d0
            do 105 k=1,nstar
                  auxil1=wvec(k)*(t(ind))
                  auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
                  if (auxil1.ge.auxil2) then
                      Anaux=Anaux+auxil1
                  else
                      Anaux=Anaux+auxil2
                  endif
105               continue
            Astarn(ind)=2.d0*Anaux/dble(nstar)
104         continue
      call tauK(t,pas,Astarn,taut3,taus,integ)
      end
**********************************************************************
 SOUS-PROGRAMME CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 50/1001
**********************************************************************
      subroutine kho2100(xvec,yvec,taut3)
      include 'commun1001'
      integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *       istar(n),k,ind,nstar,i
      real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     *       xstar(n), ystar(n), rvec(n), wvec(n),rstar21
      real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)


      real*8 taut3,taus,integ(d)
      do 199 ind=1,d
            t(ind)=dble(ind)/dble(d)
199         continue
      pas=t(2)-t(1)
      rstar21=50.d0
      nstar=0.d0
      call indexx(n,xvec,antirgx)
      call indexx(n,yvec,antirgy)
      do 101 i=1,n
            xord(i)=xvec(antirgx(i))
            yord(i)=yvec(antirgy(i))
101         continue
      call rank(n,antirgx,rgx)
      call rank(n,antirgy,rgy)
      do i=1,n
              rvec(i)=0.d0
      enddo
      do 102 i=1,n
            xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
            ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
            rvec(i)=xstar(i)+ystar(i)
            if (rvec(i).gt.rstar21) then
                              nstar=nstar+1
                              istar(nstar)=i
                          endif
102         continue
      do 103 k=1,nstar
            wvec(k)=xstar(istar(k))/rvec(istar(k))
103         continue
      do 104 ind=1,d-1
            Anaux=0.d0
            do 105 k=1,nstar
                  auxil1=wvec(k)*(t(ind))
                  auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
                  if (auxil1.ge.auxil2) then
                      Anaux=Anaux+auxil1
                  else
                      Anaux=Anaux+auxil2
                  endif
105               continue
            Astarn(ind)=2.d0*Anaux/dble(nstar)
104         continue
      call tauK(t,pas,Astarn,taut3,taus,integ)
      end
**********************************************************************
 SOUS-PROGRAMME CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 100/910
**********************************************************************
      subroutine kho1200(xvec,yvec,taut3)
      include 'commun910'
      integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *       istar(n),k,ind,nstar,i
      real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     *       xstar(n), ystar(n), rvec(n), wvec(n),rstar21
      real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)


      real*8 taut3,taus,integ(d)
      do 199 ind=1,d
            t(ind)=dble(ind)/dble(d)
199         continue
      pas=t(2)-t(1)
      rstar21=100.d0
      nstar=0.d0
      call indexx(n,xvec,antirgx)
      call indexx(n,yvec,antirgy)
      do 101 i=1,n
            xord(i)=xvec(antirgx(i))
            yord(i)=yvec(antirgy(i))
101         continue
      call rank(n,antirgx,rgx)
      call rank(n,antirgy,rgy)
      do i=1,n
              rvec(i)=0.d0
      enddo
      do 102 i=1,n
            xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
            ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
            rvec(i)=xstar(i)+ystar(i)
            if (rvec(i).gt.rstar21) then
                              nstar=nstar+1
                              istar(nstar)=i
                          endif
102         continue
      do 103 k=1,nstar
            wvec(k)=xstar(istar(k))/rvec(istar(k))
103         continue
      do 104 ind=1,d-1
            Anaux=0.d0
            do 105 k=1,nstar
                  auxil1=wvec(k)*(t(ind))
                  auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
                  if (auxil1.ge.auxil2) then
                      Anaux=Anaux+auxil1
                  else
                      Anaux=Anaux+auxil2
                  endif
105               continue
            Astarn(ind)=2.d0*Anaux/dble(nstar)
104         continue
```

```
      call tauK(t,pas,Astarn,taut3,taus,integ)
      end
**********************************************************************
 SOUS-PROGRAMME CALCULANT L'ESTIMATEUR DE KHOUDRAJI AVEC UN SEUIL DE 100/1001
**********************************************************************
      subroutine kho2200(xvec,yvec,taut3)
      include 'commun1001'
      integer rgx(n), rgy(n), antirgx(n), antirgy(n),
     *     istar(n),k,ind,nstar,i
      real*8 xvec(n),yvec(n), xord(n), yord(n),pas,
     *     xstar(n), ystar(n), rvec(n), wvec(n),rstar21
      real*8 Anaux,auxil1,auxil2,Astarn(d),t(d)


      real*8 taut3,taus,integ(d)
      do 199 ind=1,d
            t(ind)=dble(ind)/dble(d)
 199        continue
      pas=t(2)-t(1)
      rstar21=100.d0
      nstar=0.d0
      call indexx(n,xvec,antirgx)
      call indexx(n,yvec,antirgy)
      do 101 i=1,n
            xord(i)=xvec(antirgx(i))
            yord(i)=yvec(antirgy(i))
 101        continue
      call rank(n,antirgx,rgx)
      call rank(n,antirgy,rgy)
      do i=1,n
              rvec(i)=0.d0
      enddo
      do 102 i=1,n
            xstar(i)=1.d0/log(dble(n)/(dble(rgx(i))-0.5d0))
            ystar(i)=1.d0/log(dble(n)/(dble(rgy(i))-0.5d0))
            rvec(i)=xstar(i)+ystar(i)
            if (rvec(i).gt.rstar21) then
                                  nstar=nstar+1
                                  istar(nstar)=i
                              endif
 102        continue
      do 103 k=1,nstar
            wvec(k)=xstar(istar(k))/rvec(istar(k))
 103        continue
      do 104 ind=1,d-1
            Anaux=0.d0
            do 105 k=1,nstar
                   auxil1=wvec(k)*(t(ind))
                   auxil2=(1.d0-wvec(k))*(1.d0-t(ind))
                   if (auxil1.ge.auxil2) then
                       Anaux=Anaux+auxil1
                   else
                       Anaux=Anaux+auxil2
                   endif
 105               continue
            Astarn(ind)=2.d0*Anaux/dble(nstar)
 104        continue
      call tauK(t,pas,Astarn,taut3,taus,integ)
      end
**********************************************************************
        FICHIER DE DECLARATIONS DE VARIABLES COMMUNES AUX DIFFERENTS PROGRAMMES
ET
 SOUS-PROGRAMMES
**********************************************************************
      implicit none
      integer n, d, nrepli
      parameter (n=910,d=100, nrepli=50)
      real*8 r, alpha, m, parA, c, ord
      real*8 drand
      common /var1/ r
      common /var2/ m, alpha, c, ord
      common /var3/ parA
      external drand
**********************************************************************
        FICHIER DE DECLARATIONS DE VARIABLES COMMUNES AUX DIFFERENTS PROGRAMMES
ET
 SOUS-PROGRAMMES
**********************************************************************
      implicit none
      integer n, d, nrepli
      parameter (n=1001,d=100, nrepli=50)
      real*8 r, alpha, m, parA, c, ord
      real*8 drand
      common /var1/ r
      common /var2/ m, alpha, c, ord
      common /var3/ parA
      externaldrand
**********************************************************************
POUR LES PROCEDURES indexx, rank, tauK, VOIR ANNEXE E
**********************************************************************
```

IMAGE EVALUATION TEST TARGET (QA-3)



150mm

6″

APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989