MASTER EN SCIENCES INFORMATIQUES

Contrôle de qualité
analyse des méthodes de cusum et implémentation en interactif

Renoir, Vincent

Award date:
1979

Awarding institution:
Universite de Namur

FACULTES UNIVERSITAIRES NOTRE-DAME DE LA PAIX - NAMUR

INSTITUT D'INFORMATIQUE

ANNEE ACADEMIQUE 1978 - 1979

CONTROLE DE QUALITE : ANALYSE DES METHODES DE CUSUM ET IMPLEMENTATION EN INTERACTIF

PROMOTEUR : M. NOIRHOMME - FRAITURE

MEMOIRE PRESENTE PAR
VINCENT RENOIR
EN VUE DE L'OBTENTION
DU GRADE DE LICENCIE ET
MAITRE EN INFORMATIQUE

FACULTES UNIVERSITAIRES NOTRE-DAME DE LA PAIX - NAMUR

INSTITUT D'INFORMATIQUE

ANNEE ACADEMIQUE 1978 - 1979

CONTROLE DE QUALITE : ANALYSE DES METHODES DE CUSUM ET IMPLEMENTATION EN INTERACTIF

PROMOTEUR : M. NOIRHOMME - FRAITURE

MEMOIRE PRESENTE PAR
VINCENT RENOIR
EN VUE DE L'OBTENTION
DU GRADE DE LICENCIE ET
MAITRE EN INFORMATIQUE

AVANT - PROPOS

Tout mémoire est le fruit non seulement d'un travail individuel mais aussi d'une assistance de quelques personnes qui interviennent d'une manière ou d'une autre à sa bonne réalisation.
Il en est de même de celui-ci. Dès lors, je tiens en tout premier lieu à remercier :

Me M. Noirhomme-Fraiture
M. J. Demarteau , promoteurs dont les conseils furent toujours judicieux.

M. le Docteur M. Noël qui a accepté de mettre son matériel à ma disposition et qui m'a permis de consulter à loisir les données qui m'étaient nécessaires.

M. le Docteur P. Beyne qui m'a procuré toute la documentation dont il disposait quant aux différentes méthodes de contrôle de qualité.

M. J.P. Leclercq qui m'a été d'une très grande aide au démarrage de ce mémoire.

Enfin, toutes les personnes sans qui ce mémoire n'aurait pu se réaliser.

P L A N

Chapitre II

Chapitre III

III.I. Application pratique. Adaptations des méthodes pour l'implémentation

-Conclusion

-Bibliographie

-Annexes

INTRODUCTION.

Introduction au contrôle de qualité.

Dans de nombreux domaines et notamment dans les analyses médicales, la mécanisation se fait de plus en plus envahissante. Vu la rapidité gagnée grâce aux machines, on possède une grande quantité de résultats à traiter en temps réel.

C'est là que l'informatique intervient. Les diverses méthodes d'analyse des résultats se doivent de posséder trois qualités essentielles qui sont:
- fiabilité
- praticabilité
- efficacité.

La fiabilité s'apprécie entre autres par la précision (qu'il faut définir,exprimer et qu'il faudra mesurer), l'exactitude (qui va servir à la vérification des résultats) et par des critères fonctionnels tels que sensibilité, sélectivité et spécificité.

La praticabilité est essentielle: on n'est pas en effet confronté à des problèmes théoriques mais bien ici à des problèmes pratiques qu'il faut résoudre le plus vite et le mieux possible.

L'efficacité est importante car il faut éliminer la plus grande part d'aléatoire pour posséder à l'arrivée des résultats plausibles et dignes de confiance.

Deux types d'erreurs peuvent se glisser dans les processus d'analyse :

- erreur systématique : elle dépendra du choix de la méthode, des étalons et calibrations utilisés et des défections des appareils.

- erreur aléatoire : elle existe toujours mais il faut s'en méfier à cause de sa propriété d'additivité qui peut conduire à la catastrophe !

Dans l'ensemble, ce que l'on appelle contrôle de processus fera intervenir la méthode utilisée qui de ce fait sera elle-même testée pour ce qui est de ses composantes (par exemple en faisant varier la concentration, la température,...). On pourra ainsi vérifier l'hypothèse de linéarité inhérente à chaque méthode quant à certains facteurs. C'est dire que le contrôle devra se faire à partir de plusieurs analyses effectuées différemment et avant de tester les résultats, c'est d'abord la méthode qui l'est !

Il en est tout autrement du contrôle de qualité à proprement parler. En effet, on ne tient plus compte que des résultats obtenus en considérant que le processus s'est déroulé de façon tout à fait normale sous des conditions optimales.

C'est à dire que l'on ne s'intéresse plus au contenu intrinsèque de la méthode utilisée pour faire les analyses mais seulement aux résultats qu'elle a fournis. Si bien que l'on ne teste plus la méthode elle-même mais bien le bon fonctionnement de la machine (dont le réglage est très fin et peu fiable).

On va procéder par une étude chronologique des résultats passés (avec leurs caractéristiques statistiques telles que moyenne et écart-type) pour décider si oui ou non le résultat courant possède des garanties de véridicité. Pour ce faire,on va effectuer régulièrement des analyses sur des sérums, solutions titrées à l'avance, dont on connaît à l'avance le résultat que l'on devrait obtenir en cas de bon fonctionnement de la machine (phénomène de bonne reproductibilité). Ces solutions de sérums vont être incorporées au hasard parmi les analyses de patients et on pourra ainsi détecter plus ou moins rapidement selon la méthode utilisée les déviations " graves " de la machine .

Le contrôle de qualité permet ainsi de détecter près de 95 % des erreurs (les 5 % restants représentent les erreurs moins graves). Pour essayer de trouver les erreurs restantes, il faudrait alors tester d'autres résultats en faisant intervenir ceux des patients soit par double analyse (assez coûteux) soit par étude de la fonction de répartition de la population entraînant un certain contrôle de qualité sur ces résultats. Etant donné le peu de normalité que montrent ces populations, ce procédé est très complexe!

Un moyen plus simple serait de comparer la moyenne journalière avec celles obtenues précédemment car on a remarqué que l'ensemble des moyennes journalières était plus ou moins normal. Ce procédé est plus fiable mais il implique une manipulation constante des données de patients entraînant un grand nombre d'opérations rendant nécessaire un matériel de test très important et très rapide.

Il en ressort donc que l'élément essentiel est fourni par le contrôle de qualité basé sur des sérums.
Il existe des sérums de deux types:

- pool de sérum: préparé à partir d'excédents journaliers recueillis et congelés.
- sérum commercial: naturel ou synthétique, d'origine humaine ou animale, titré ou non.

On utilisera,nous,du sérum commercial titré à l'avance!

CHAPITRE I

METHODES FONDAMENTALES DE CONTROLE DE QUALITE

- Méthode de Shewart
- Méthodes utilisant des sommes cumulées
  - V- écran de Ewan & Kemp
  - V- écran de Goldsmith & Whitfield
  - Intervalle de décision Page, Ewan & Kemp.

I.1. SHEWART

L'évolution industrielle apparue au vingtième siècle a entraîné une obligation de contrôle très rapide des résultats obtenus dans divers domaines (processus de fabrication, analyses,...);ses applications principales résident dans le contrôle industriel de la qualité où les résultats provenant des essais et inspections de la production sont reçus constamment et où une décision rapide est nécessaire lorsque le processus envisagé annonce un dérèglement.
Les principes du contrôle de qualité ont tout de suite été basés sur des diagrammes. Ils ont été introduits en Grande Bretagne par Dudding et aux Etats Unis par Shewart vers la fin des années mille neuf cent vingt. Il est à remarquer que les méthodes actuelles reposent aussi essentiellement sur des diagrammes (seulement, on utilisera des CUSUM, diagrammes de sommes cumulées.)

Le système de Shewart et Dudding repose sur une population de moyenne M et d'écart-type S connus. Ceci suppose bien entendu que l'on dispose au départ d'une certaine quantité de données (cinquante à cent observations peuvent suffire) pour pouvoir se donner une idée assez proche de la réalité des paramètres estimés. Dans certains cas, on pourra se fixer à l'avance le niveau moyen M comme quantité de référence à laquelle on voudrait aboutir comme moyenne des résultats à obtenir. On peut donc tester les données par rapport à cette moyenne fictive et une fois le niveau atteint reprendre la moyenne des derniers résultats comme ligne de conduite.

C'est donc la moyenne qui va servir de ligne centrale du diagramme en vue de contrôler la moyenne du processus. Les limites pour une action corrective sont fixées à $M \pm 3.09\ S$.

On reporte sur le diagramme les résultats au fur et à mesure de leur disponibilité. Aussi longtemps que les points se situent à l'intérieur des limites, on admet que le système se déroule normalement, tandis que l'apparition d'un point en-dehors des limites entraîne une action corrective car il est l'indice d'une défaillance. Là s'arrête la tâche du processus de contrôle de qualité: situer et corriger la cause de l'erreur n'étant pas de son ressort.

Pourquoi placer les limites d'intervention à $M \pm 3.09\ S$?

On les place de cette manière pour que, si la moyenne du processus se maintient effectivement au niveau M, la probabilité qu'un résultat tombe en-dehors des limites soit de 0.002 c'est-à-dire qu'une action corrective soit déclenchée à tort une fois sur cinq cents en moyenne (si l'on suppose que la distribution des résultats est normale).

Rem. On peut aussi utiliser cette méthode pour détecter des changements de valeur de l'écart-type d'une population (cf Statistical Methods in Research and Production).

Le principal avantage qu'offrait la méthode de Shewart était sans nul doute sa très grande simplicité d'application, la clarté de visualisation que procurait le diagramme ainsi que la rapidité à détecter les grandes déviations par rapport à la moyenne. Par contre, il est impossible par ce procédé de détecter rapidement des variations de la moyenne courante inférieure à trois écarts-types (une déviation constante de un écart-type n'étant détectée en moyenne qu'après un report de 55 résultats sur le diagramme).

On peut tout de même améliorer la méthode en ajoutant d'autres lignes de confiance à $M \pm I.96\ S$, en incluant la règle que 2 résultats consécutifs en-dehors de ces nouvelles limites dénotent un changement dans la valeur moyenne du processus.

Il existe encore d'autres améliorations telles que :

a) ajouter les limites M ± S, trois points consécutifs en-dehors du couloir suffisant pour entraîner une intervention corrective.

b) ajouter la règle supplémentaire suivant laquelle onze points consécutifs du même côté de la moyenne montrent un abaissement ou une élévation de celle-ci.

c) ajouter des règles du type: " si les m derniers résultats sont du même côté de la moyenne, intervenir si l'un de ces résultats se trouve en-dehors de M ± bS où b est un paramètre dépendant de m."

Tous ces procédés groupés assurent que le risque d'intervention injustifiée est toujours d'un peu plus de un sur cinq cents mais un changement peu important par rapport à la moyenne sera détecté beaucoup plus rapidement. Il va de soi par contre que si la principale qualité de la méthode initiale de Shewart était la simplicité et la visualisation, il n'en est plus de même!

Un problème rencontré par ces méthodes est qu'elles n'utilisent pas toutes les données qu'elles ont à leur disposition mais seulement les dernières données courantes et la région dans laquelle elles se situent. C'est dire qu'il semble qu'il y ait une certaine perte d'efficacité en ne considérant que la région où la donnée individuelle se situe au lieu de retenir principalement sa valeur numérique effective .

C'est ce qui va être résolu par les méthodes utilisant les sommes cumulées.

Pour un exemple numérique, se reporter à l'annexe I, pages 1 et 2 où l'on pourra remarquer la lenteur manifeste du procédé initial de Shewart, l'amélioration apportée par les nouvelles règles et malgré cela sa relative lenteur par rapport aux méthodes utilisant les sommes cumulées.

I. 2. Procédés utilisant des diagrammes de sommes cumulées.

I.2.1. Description des diagrammes

Précédemment, les diagrammee étaient établis en portant en abscisse les numéros de variables et en ordonnée leurs valeurs numériques. Au lieu de prendre cette représentation, on choisit de mettre en ordonnée des sommes cumulées, c'est-à-dire des sommes du type : $S_r = \sum_{i=1}^{n} (x_i - k)$ où k sera une valeur choisie au départ par exemple égale à l'objectif à atteindre. Souvent, il sera égal à la moyenne, ce qui permettra de se faire une idée de l'écart à la moyenne que le processus impliquera au fil du temps.

Pour la clarté du diagramme, il est important de bien choisir les unités. Si on considère la distance horizontale entre deux points comme étant une unité, il est recommandé que la même distance sur l'échelle verticale représente en fait $\pm$ 2 S (où S est l'écart-type à court terme de la série).

Avec ce système d'échelles, l'accroissement moyen aura une pente de 45° pour une différence de 2S par rapport à la moyenne k et une variable aléatoire aura une très bonne représentation . On peut ainsi faire varier l'échelle verticale s'il est très important de détecter une déviation connue, par exemple Δ.

Il faut donc choisir les échelles t.q.

$$\Delta < \frac{\text{longueur d'une unité sur échelle horiz.}}{\text{longueur d'une unité sur échelle vert.}} < 2\Delta$$

Tout nombre entier satisfaisant à cette condition est acceptable.

Ce genre de diagramme est très intéressant en ce sens qu'une petite déviation de la moyenne va faire croître la courbe très rapidement; la pente de celle-ci nous donnera le sens de variation tandis qu'une seule observation farfelue ne

modifiera pas fortement la pente de la courbe mais sera pris comme point évident de discontinuité.
Avec ce diagramme, on dispose en plus d'une représentation du processus s'étendant au dernier point, aux 2 derniers points, etc..., aussi loin que nous le voulons. On possède ainsi une idée de la dérive du processus par le calcul de la pente de la droite joignant le $n^{è}$ au $(n-I)^{è}$ point, le $n^{è}$ au $(n-2)^{è}$, ... et ainsi de suite.

Exemple numérique : voir annexe I pages 3 et 4

Si l'on examine de plus près le diagramme cumulatif, on remarque que si le cheminement est plus ou moins horizontal, le processus garde environ la même moyenne ou en tout cas, est en état de contrôle statistique autour de k. Si le tracé s'élève, c'est que la moyenne du processus a augmenté et inversément.

Il suffit donc maintenant de se fixer une limite d'accroissement (positif ou négatif) pour pouvoir effectuer un contrôle de qualité sur les résultats considérés. La règle à adopter est alors d'intervenir si le point courant du diagramme s'élève (s'abaisse) de plus d'une hauteur donnée h au dessus du point le plus bas (haut) enregistré sur le diagramme cumulatif depuis la dernière fois qu'a été prise une telle décision.

Cette manière de procéder a été proposée par Page, améliorée par Ewan et Kemp. Un autre procédé tout à fait équivalent a été introduit par Barnard : il s'agit d'un écran en forme de V. Ces différentes méthodes vont être développées et analysées dans les pages qui suivent.

I.2.2. Méthode du masque en V ou V-écran

Description

La méthode du masque en V ou V-écran est équivalente à celle d'un schéma simple avec deux limites horizontales simultanées. L'avantage se situe dans le fait qu'il est possible de déterminer les limites à partir des résultats que l'on veut obtenir.

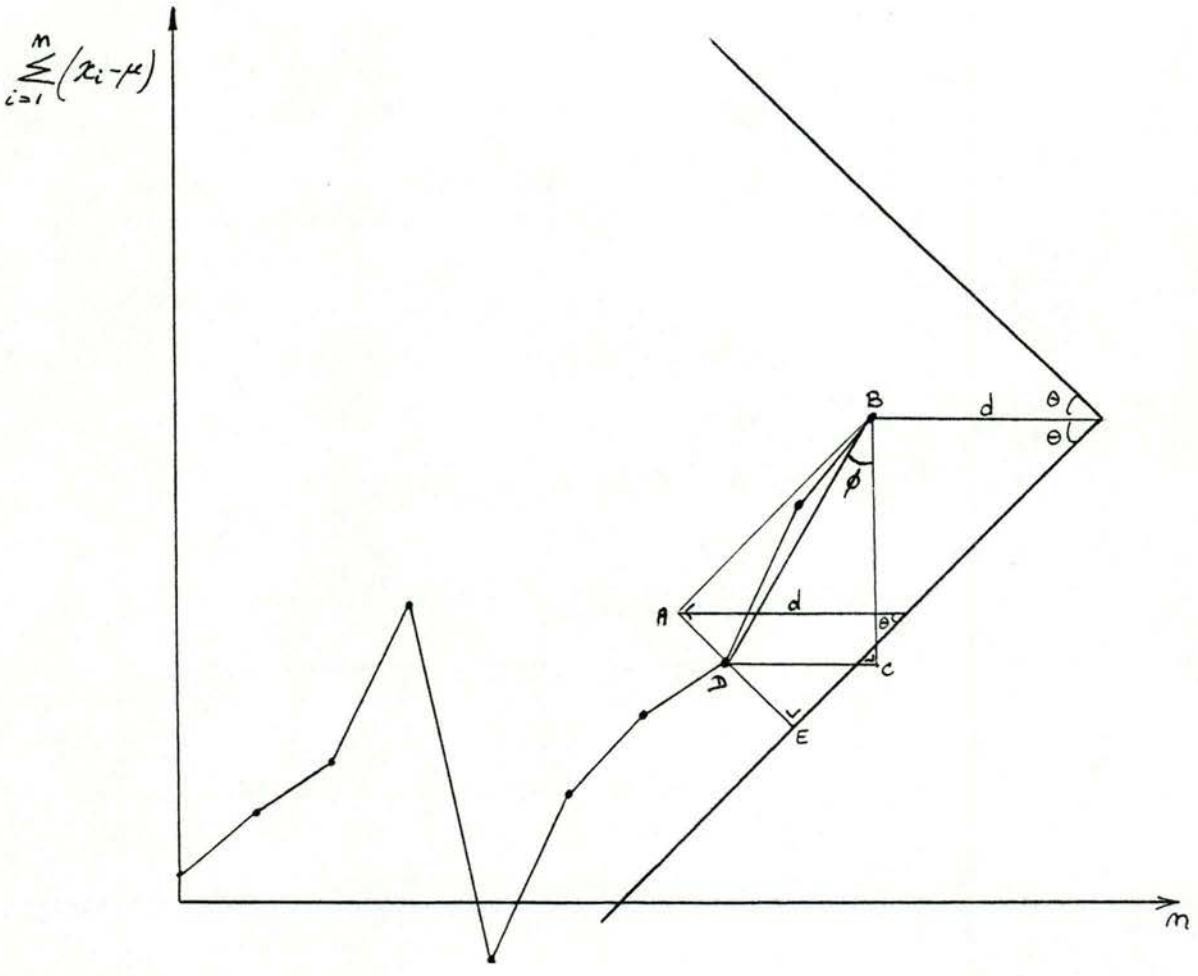
On définit ce que l'on appelle ARL (§) ("average run length") : c'est le nombre de données à tester avant intervention corrective quand le processus s'éloigne de § de la moyenne.

Il est alors possible, après avoir choisi un ARL(0) et un ARL(2S), de construire un V-écran satisfaisant nos exigences. On utilise un diagramme de sommes cumulées où les ordonnées sont du type $S_m = \sum_{i=1}^{m} (x_i - \mu)$

où μ est la valeur cible (pas nécessairement la moyenne), c'est-à-dire que la courbe des points sera plus ou moins horizontale quand le processus variera autour de μ .

Exemples de V-écrans : voir annexe I pages 5 et 6

Le V-écran est un angle symétrique par rapport à l'horizontale et dont le sommet se trouve à une distance d du dernier point reporté et qui, en cas de stabilité du processus, englobe tous les points du diagramme. L'ouverture du V-écran est représentée par 2 θ .



On se propose de construire un V-écran et de dégager du schéma les conditions d'inacceptabilité des données inhérant au choix de d et θ .

Explication de la méthode

$DC = BD \sin \phi$

$C\hat{B}A = \pi/2 - \theta$ $\qquad$ $C\hat{B}D = \phi$

$AD = BD \cos(\theta + \phi)$

$= BD \cos\theta \cos\phi - BD \sin\phi \sin\theta$

$= BC \cos\theta - DC \sin\theta$

Rem. $\theta < 90°$ car sinon, aucun contrôle ne serait plus possible : $\rightarrow \cos\theta \neq 0$.

$$\frac{AD}{\cos\theta} = BC - DC \operatorname{tg}\theta$$

Or un point sera en-dehors de l'écran si

$AD \geqslant AE$ or $AE = d \sin\theta$

d'où $AD \geqslant d \sin\theta$

où encore $\dfrac{AD}{\cos\theta} \geqslant d \operatorname{tg}\theta$

on aura donc : $BC - DC \operatorname{tg}\theta \geqslant d \operatorname{tg}\theta$

$$\begin{cases} DC = r\,w \\ BC = \sum_{i=m-n}^{m} (x_i - \mu) \end{cases}$$

Il y aura donc dépassement vers le bas quand

$$\exists\, r \text{ t.q. } \sum_{i=m-n}^{m} (x_i - \mu) - \sum_{i=m-n}^{m} w \operatorname{tg}\theta \geqslant d \operatorname{tg}\theta$$

$$\text{ou encore } \sum_{i=m-n}^{m} (x_i - \mu - w \operatorname{tg}\theta) \geqslant d \operatorname{tg}\theta$$

Similairement, il y aura dépassement vers le haut si

$$\exists\, r \text{ t.q. } \sum_{i=m-n}^{m} (X_i - \mu + w \operatorname{tg}\theta) \leqslant -d \operatorname{tg}\theta$$

L'usage du V-écran est donc équivalent au calcul d'une double somme et d'intervenir s'il existe un r qui vérifie l'une des deux inéquations.

Longueurs moyennes de séries

Voyons maintenant comment calculer les longueurs moyennes de séries (ARL) correspondant au processus.

Posons : w t g θ = valeur de référence = k

d tg θ = intervalle de décision = h

$y_i = x_i - \mu$

Considérons les sommes :

$$S_j = \sum_{i=t}^{t+j} (y_i - k) \qquad s_\ell = \sum_{i=\mu}^{\mu+\ell} (y_i + k)$$

Considérons maintenant S_m et s_n t.q.

$0 < S_j < h$ pour $0 \leqslant j \leqslant m$ (a)

$0 > s_\ell > -h$ pour $0 \leqslant \ell \leqslant n$ (b)

Si les sommes cumulées des deux schémas se trouvent entre les bornes, alors S_m et s_n vont les représenter si

$$m + t = n + u = r$$

Que se passe-t-il si $S_{m+1} \geqslant h$

Alors $S_{m+1} = S_m + y_{m+1} - k \geqslant h$

ou encore $y_{m+1} \geqslant h + k - S_m$

si bien que $s_{n+1} = s_n + y_{n+1} + k \geqslant s_n - S_m + h + 2k$

$$\simeq \begin{cases} s_{t-u-1} + h + 2(m+2)k \ \ldots\ldots\ldots (t > u) \\ h + 2(n+2)k \ \ldots\ldots\ldots\ldots\ldots (t = u) \\ h + 2(n+2)k - S_{u-t-1} \ \ldots\ldots\ldots (t < u) \end{cases}$$

c'est-à-dire que $s_{n+1} > 0$ par (a) et (b).

De même, on peut montrer que si $s_{n+1} \leqslant -h$, alors $S_{m+1} < 0$.

Par ce fait, on en arrive à la conclusion que si un point se trouve dans les limites choisies, il n'est pas possible que le point suivant fasse franchir en même temps les deux branches du V-écran. Ceci entraîne l'indépendance des deux processus (sortie par le haut et sortie par le bas) au point de vue longueur moyenne des séries.

Donc, pour calculer un certain ARL du V-écran, soit (L), il suffit de le calculer pour le premier processus, soit (L_1), pour le second processus, soit (L_2), puis de résoudre

$$\frac{1}{L} = \frac{1}{L_1} + \frac{1}{L_2} \qquad (c)$$

Pour calculer L_1 et L_2, on peut utiliser les résultats de Ewan et Kemp (1960) qui les ont calculés pour une grande variété de schémas. Ces résultats ont été utilisés pour calculer les ARL de V-écrans de dimensions données.
Les ARL ainsi calculées ont été reportées sur nomogrammes.

Remarque : On peut avoir un V-écran non symétrique par rapport à l'horizontale si par exemple, on veut détecter plus vite une augmentation qu'une diminution de la moyenne.
On peut alors montrer que l'équation (c) reste tout à fait correcte si
$w \sin(\theta_1 + \theta_2) > d \sin(\theta_1 - \theta_2)$ $\qquad \theta_1 > \theta_2$

Avec un degré raisonnable d'approximation, on peut aussi admettre que l'équation (c) reste valable pour un grand nombre de valeurs de θ_1 et θ_2 qui ne satisfont pas cette dernière équation.

V-écran pour variables normales

S'il est possible de calculer les ARL quand on connaît les paramètres d et θ , il est également possible d'effectuer le passage inverse en utilisant ces mêmes nomogrammes d'Ewan et Kemp. La table ci-après montre des valeurs obtenues par ce principe.
On obtient ainsi les dimensions d'un certain nombre de V-écrans en partant des résultats que l'on voudrait obtenir au point de vue longueurs moyennes de séries.

ARL($\Delta\mu$)	ARL(0)		
	100	250	500
3	2.27 0.91	2.20 1.03	2.11 1.13
4	3.26 0.76	3.24 0.85	3.19 0.92
5	4.40 0.65	4.30 0.74	4.26 0.80
6	5.57 0.58	5.36 0.66	5.23 0.72
7	6.64 0.52	6.33 0.60	6.28 0.65
8	7.76 0.48	7.50 0.55	7.27 0.60
9	8.84 0.44	8.43 0.51	8.19 0.57

le premier nombre = d / w

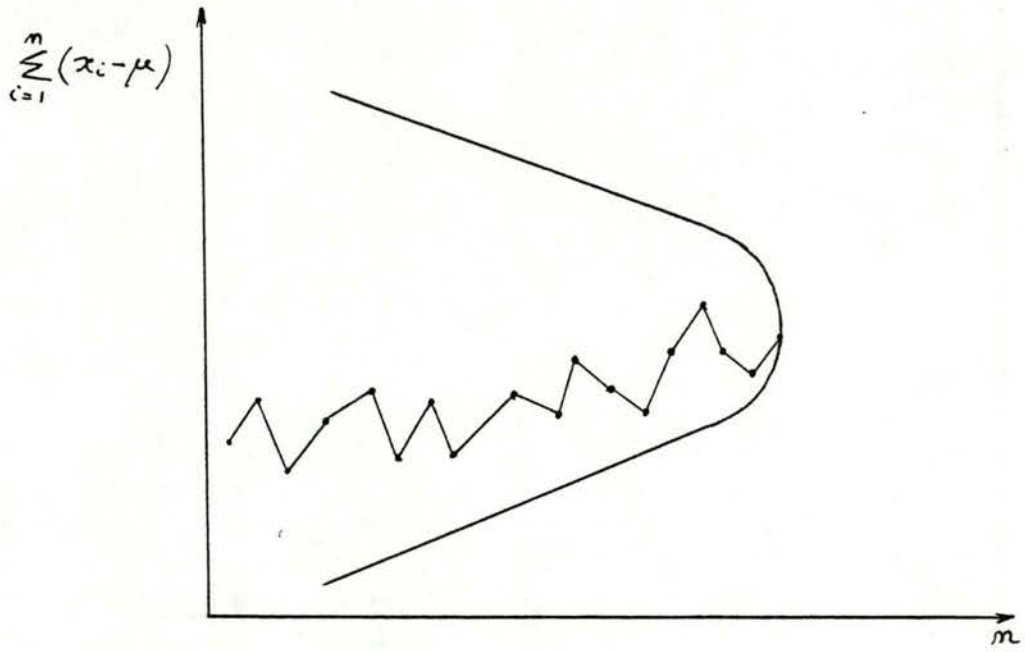
le deuxième nombre = $\frac{\Delta\mu}{2S}$

$$\theta = tg^{-I} \left(\frac{\Delta\mu}{2w} \right)$$

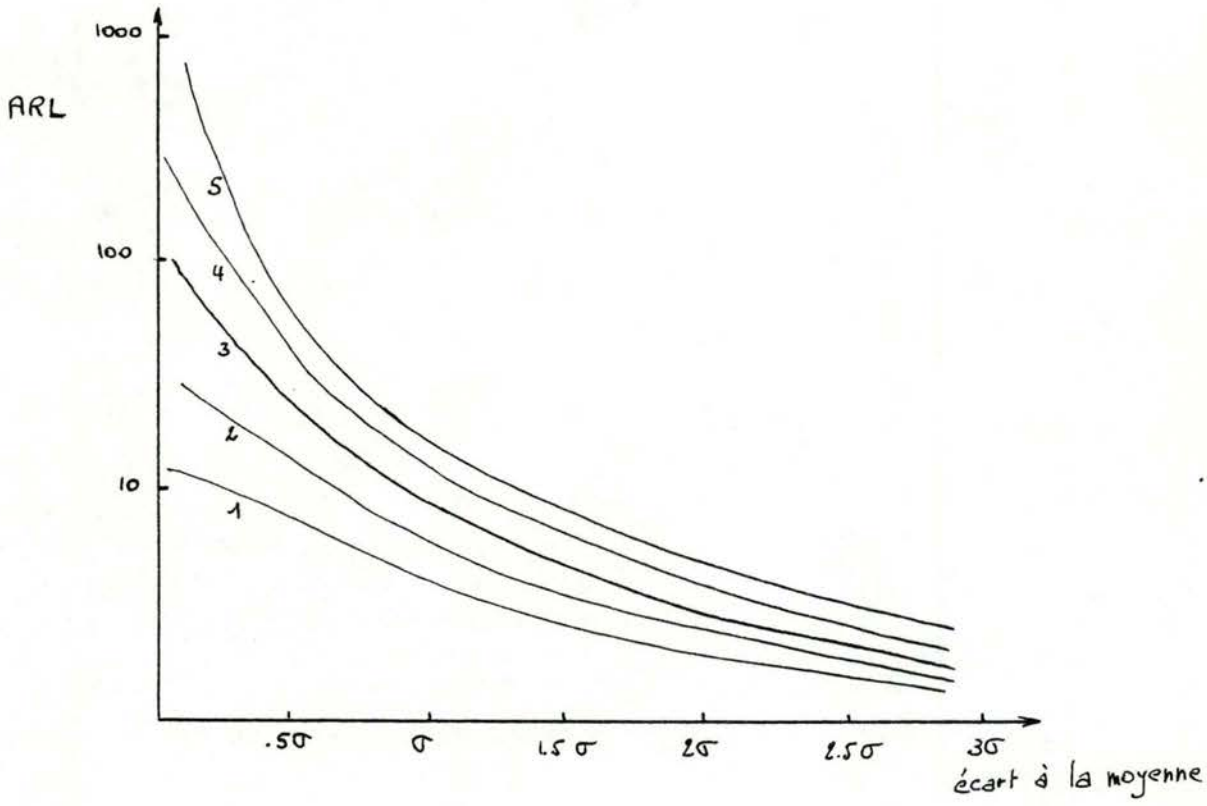
I.2.3. Méthode du V-écran modifié

Description

Etant donné la lenteur avec laquelle réagit le V-écran quand se manifeste un brusque changement dans le processus, on peut modifier le V-écran et le remplacer par une parabole ou V-écran modifié. Le schéma se présente alors comme suit :



Les ARL sont alors calculés par simulation. On génère des nombres pseudo-aléatoires, on transforme la série en série normale par la méthode Box-Muller. En y appliquant les déformations nécessaires, on calcule alors les différents ARL. On obtient des diagrammes du type :



Etant donné le faible intérêt que nous portons quant à la recherche rapide de brusques et brefs changements dans la valeur moyenne d'un processus en ce qui concerne le contrôle de qualité, on renverra le lecteur pour une étude plus approfondie à : "Technometrics, Vol. 15, N° 4, nov.73

A Modified V-Mask Control Scheme.

James M. Lucas"

Première comparaison des trois méthodes déjà analysées.

Méthodes	0	S/2	S	2S	3S	4S	5S ← ECARTS
Shewart	320	137	39.5	5.89	1.92	1.17	1.02
V-écran modifié	320	54.2	10.6	3.37	1.73	1.29	1.08
V-écran	319	42.3	9.23	3.50	2.09	1.59	1.21 ← ARL

On voit donc que la méthode de Shewart met beaucoup de temps à réagir pour un écart de S ou moins par rapport à la moyenne mais est assez rapide pour un écart supérieur à 3S.

Le V-écran modifié quant à lui, est un peu plus rapide que le V-écran à partir d'un écart supérieur à 2 ou 3 écarts-types mais plus lent pour un plus petit écart.
Les qualités du V-écran semblent maintenant suffisantes pour que l'on puisse envisager son application au cas qui nous intéresse.

I.2.4. Autre méthode de détermination des paramètres d et θ

Le choix des paramètres d et θ est primordial pour la bonne marche du contrôle par V-écran. En effet, plus grandes sont l'importance du décalage d et l'ouverture θ du V, plus rares seront les interruptions du processus.
Une façon de choisir un V-écran convenant à un processus est de superposer au diagramme plusieurs V-écrans et d'ajuster la forme de l'écran de façon à ce que les changements observés soient détectés dans un délai raisonnable (sans tenir compte des fluctuations aléatoires du processus).

Procéder de telle sorte demande évidemment beaucoup de manipulations et n'est donc pas à conseiller.

De nouveau, le plus simple est de partir des longueurs moyennes de séries (ARL) quand le processus est autour de μ (L_0) et quand il s'en éloigne de $\Delta\mu$ (L_1). On exigera généralement de L_0 qu'il atteigne la centaine ou plus tandis qu'on acceptera un L_1 entre 3 et 10.

Ces longueurs moyennes de séries ont été estimées par Goldsmith et Withfield pour un certain nombre de V-écrans en utilisant les méthodes de Monte-Carlo pour des populations normales $N(\mu, \sigma)$. Ils ont ensuite porté les résultats sur des courbes.

On trouvera à l'annexe 1 pages 7 et 8 les diagrammes résultant des simulations et fournissant un choix considérable de V-écrans selon la distance d.

A partir de L_0 et L_1, il suffit donc de choisir la courbe donnant les résultats les plus proches : on possède alors d et tg θ . On peut construire le V-écran.

Remarque : Les effets d'une corrélation de série (assez fréquente) diminuent l'efficacité du dispositif qui lui,a été développé pour une variable tout à fait normale. On en reparlera dans les aspects statistiques au chapitre II.

I.2.5. Dispositif basé sur l'intervalle de décision

Page, Ewan, Kemp.

Description

Ce dispositif va jouer sur la variation de la somme cumulée depuis son dernier extrêmum. Il ne nécessite aucun écran ou même de diagramme. La façon de procéder est d'intervenir lorsque le point courant du diagramme dépasse de plus d'une grandeur h le point antérieur le plus bas ou le point antérieur le plus haut.

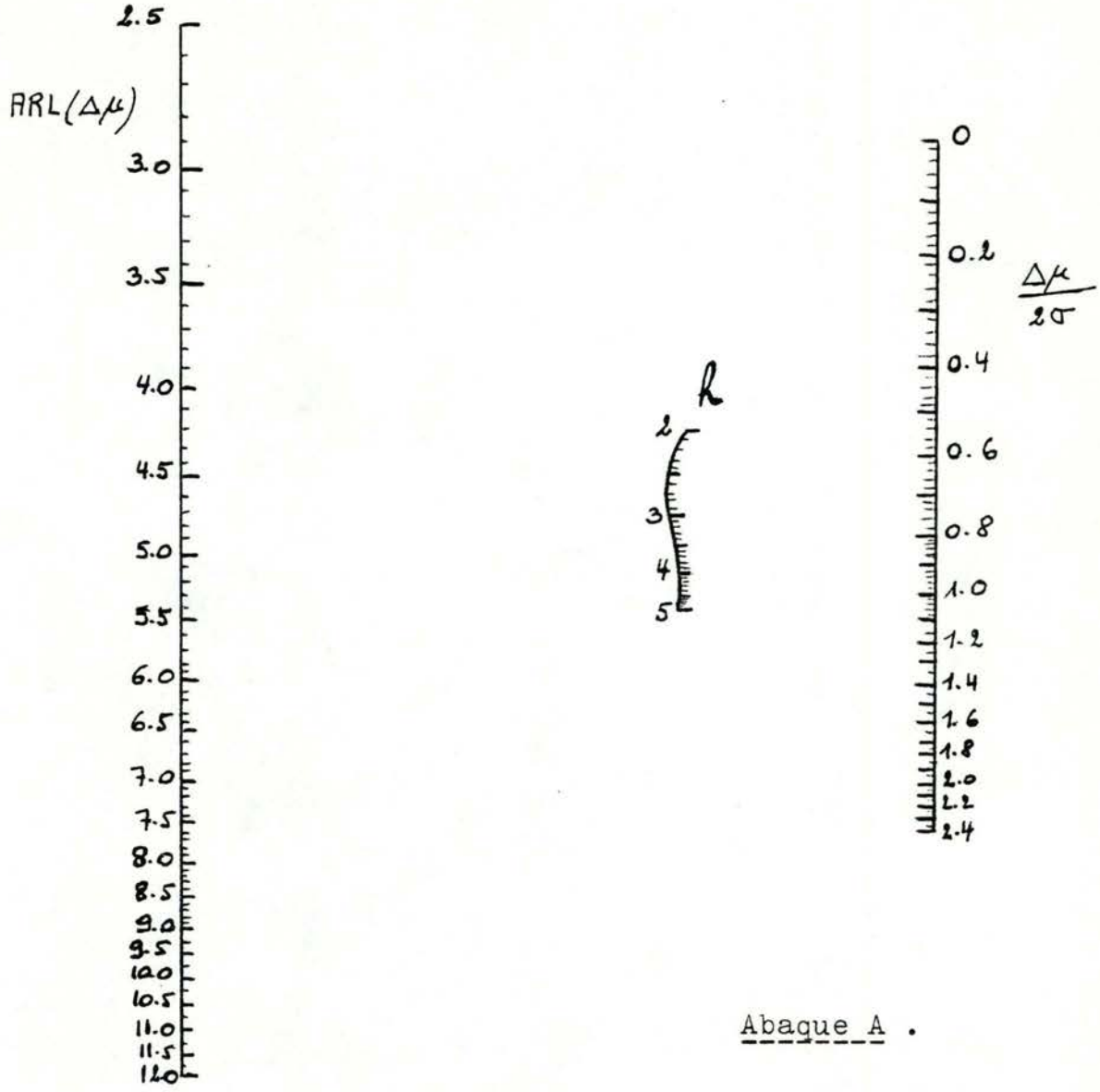
La quantité h est dite "intervalle de décision".
Dans le cas où l'on ne veut détecter qu'une augmentation, on fixe la valeur centrale k un peu au dessus de la moyenne, c'est-à-dire à mi-chemin entre la moyenne et le niveau qui apparaît comme juste insatisfaisant. On ne représentera donc la somme cumulée que pour les valeurs situées au dessus de k.

Exemple numérique : annexe 1 pages 9,10.

Il suffit pour effectuer un contrôle de qualité avec cette méthode de considérer une somme cumulée, la quantité h et les deux niveaux les plus bas et les plus hauts. Il reste alors à comparer les différences entre le dernier point courant et les deux niveaux extrêmes avec l'intervalle de décision h.

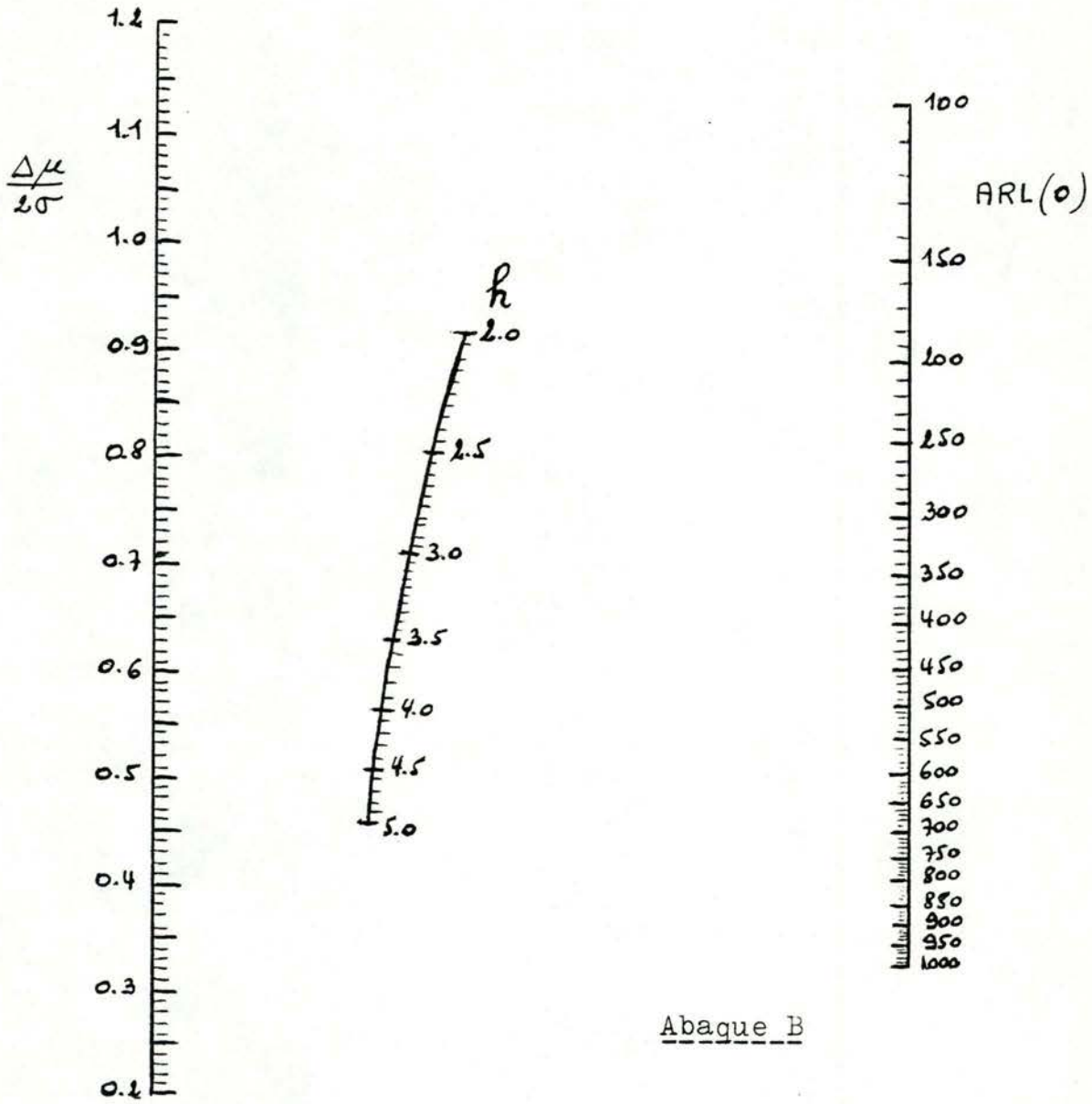
Longueurs moyennes de séries

Elles ont été calculées par Ewan et Kemp dans le cas où les résultats sont distribués normalement. Elles peuvent être tirées des abaques A et B de la façon suivante.
On possède au départ un objectif à atteindre sous forme d'ARL à ne pas dépasser. A partir de cette valeur, on détermine les autres paramètres qui nous sont nécessaires.



Abaque A .

Supposons que l'on veuille ne pas dépasser un certain ARL ($\Delta\mu$). On reporte cette valeur à gauche sur l'abaque A. A droite, on porte $\frac{\Delta\mu}{2\sigma}$. La droite joignant les deux points détermine la valeur de h à l'intersection avec la courbe.
On reporte ensuite sur l'abaque B les valeurs $\frac{\Delta\mu}{2\sigma}$ et h ; on trouve ainsi l'ARL (0) correspondant.



Abaque B

Remarque : Si les points envisagés comme résultats sont en fait des moyennes d'échantillons de taille n et d'écart-type s', il convient de multiplier les valeurs h et $\frac{\Delta\mu}{2\sigma}$ par $\frac{\sqrt{n}}{s'}$.

Equivalence avec dispositif utilisant les V-écrans.

En raison de la similitude dans leur mode d'élaboration, on peut montrer que le V-écran et le dispositif basé sur l'intervalle de décision sont exactement équivalents : il suffit de considérer la relation: $h = 2\sigma\, d\, tg\theta$

Cette propriété est très importante car elle va nous permettre d'utiliser la même méthode quand il s'agira d'effectuer un contrôle de façon numérique. C'est seulement la valeur des paramètres qui va changer selon la méthode utilisée.

I.2.6. Choix d'un procédé de contrôle.

Les procédures utilisant les intervalles de décision exigeant un minimum de diagrammes seront de ce fait souvent utilisées. Par contre le V-écran visualise beaucoup mieux l'évolution du processus indépendamment de l'émission de signaux d'intervention. Quoi qu'il en soit, ces deux méthodes restent toujours supérieures à celle de Shewart et cela principalement grâce à l'utilisation des sommes cumulées.
C'est dire que dans notre cas, il faudra choisir un ou plusieurs procédés à analyser. Le choix va donc porter sur d'autres critères relatifs à des caractéristiques de population.
Ces critères de choix seront expliqués dans les chapitres II et III. Il nous permettront de retenir une solution acceptable tant au point de vue statistique que sur le plan pratique.

CHAPITRE II

POINT DE VUE STATISTIQUE

- Normalité ou non-normalité ?
- Corrélation sérielle
- Taille des échantillons
- Critères de choix, fiabilité

II 1. Introduction

Il s'agit en premier lieu de voir si les données dont on dispose possèdent une distribution normale.
Si c'est le cas, on peut alors appliquer sans restriction les méthodes expliquées dans le chapitre précédent.
Sinon, il s'agit de voir quelles influences apporte une non normalité de la population.

De même, les méthodes sont construites pour des données indépendantes. Que se passera-t-il si le coefficient de corrélation à l'intérieur d'une série de données est différent de zéro ?

Il va de soi que ces deux caractéristiques n'empêchent nullement d'appliquer les différentes méthodes mais elles risquent d'avoir un effet néfaste sur les longueurs moyennes de série (ARL) obtenues en pratique par rapport à celles voulues théoriquement. Les performances des méthodes sont dès lors fortement diminuées et ne correspondent certes plus à ce que l'on désirait au départ.

Les méthodes proposées faisant intervenir la moyenne et l'écart-type du processus, il s'agira donc de les évaluer le plus justement possible. Le nombre de données à envisager avant de faire un premier contrôle de qualité est donc très important dans l'erreur que l'on fera dans l'estimation de ces paramètres (puisque dans le calcul des intervalles de confiance pour la moyenne et l'écart-type le nombre de données constituant l'échantillon intervient au dénominateur ; ainsi plus on aura de données, plus l'intervalle sera petit !) .

Il faudra aussi rechercher des critères de choix d'une solution au point de vue théorique. On pourra ainsi dégrossir le problème de la fiabilité des méthodes et de la subjectivité qu'elles peuvent entraîner. On pourra dès lors faire un premier pas vers la compréhension du problème de l'introduction de l'interactif dans le contrôle de qualité.

II.2. Normalité de la population

Si la distribution est autre que normale, la valeur des longueurs moyennes de séries peut considérablement varier.

Pour des distributions non normales symétriques, l'ARL (0) n'est que légèrement affecté. Par contre, lorsque le processus se maintient dans l'intervalle $\mu \pm \sigma$, l'effet peut être plus important et ainsi modifier les ARL calculés. Si la distribution présente plus de résultats en dehors des limites $\mu \pm 3\sigma$ que la distribution normale, les longueurs moyennes de séries se trouvent réduites (c'est ainsi que ARL(0) pourrait se trouver réduit à moins du tiers de sa valeur dans le cas normal). Par contre une plus forte proportion de résultats se situant à l'intérieur des limites $\mu \pm 3\sigma$ va faire augmenter les longueurs moyennes de séries. Dans des cas extrêmes, ARL(0) peut ainsi se trouver accru d'environ quatre fois.

Si la distribution est dissymétrique, les longueurs moyennes de séries seront peu modifiées pour les grands écarts (de l'ordre de $2\text{-}3\sigma$), mais à d'autres niveaux les différences peuvent être importantes.
Si la dissymétrie est positive, c'est-à-dire plus de données supérieures à la valeur de référence que d'inférieures, on devra généralement s'attendre à ce que le procédé soit plus efficace pour déceler les dépassements par excès que les dépassements par défaut. Mais si le processus reste au voisinage de la valeur objectif, les effets seront incertains.

Si la distribution est binomiale ou de Poisson, Ewan et Kemp ont donné des ensembles de tables où l'on peut trouver l'intervalle de décision h pour les niveaux acceptables et inacceptables du paramètre de Poisson correspondant à des ARL égales à 500 au niveau acceptable et à 7 au niveau inacceptable (idem 500, 3).

Mais cette technique ne s'applique plus au contrôle de qualité proprement dit ; elle s'applique plutôt au contrôle des pourcentages faibles (Poisson) ou des nombres de survenances (binomiale).

II.3. Corrélation sérielle

En théorie, on se permet toujours de considérer les données que l'on a à notre disposition comme étant entièrement indépendantes. En pratique, il en est souvent tout autrement et généralement les observations recueillies seront sériellement corrélées.
On ne peut donc plus se fier à l'hypothèse d'indépendance faite au départ.

Il reste à voir, et c'est important, si cette corrélation va influencer le comportement des longueurs moyennes de séries.

Soient des variables pseudo-aléatoires x_i suivant une loi de distribution $N(k\sigma , \sigma')$

$$x_i = k\sigma + y_i$$

$$\text{avec} \quad y_i = a\, y_{i-1} + (1 - a^2)^{\frac{1}{2}} \varepsilon_i \qquad i = 1, 2, \ldots$$

où ε_i et y_0 sont des variables pseudo-aléatoires $N(o, \sigma^2)$

et $a < 1$

On a dès lors : $\operatorname{cov}(y_r , y_{r-s}) = a^s \sigma^2$

donc $\operatorname{corr}(y_r , y_{r-s}) = a^s \qquad s = 0, 1, 2, \ldots, r$

a = 0

On obtient donc un coefficient de corrélation nul.
On est donc dans le cas où les données sont indépendantes et l'on retrouvera de ce fait les solutions antérieures.

a = 1

Le coefficient de corrélation est égal à un. Les données ne sont donc pas indépendantes.

On a : $x_i = k\sigma + y_0 \quad \forall i$

$$S_n = n\,(k\sigma + y_0)$$

On obtient alors un ARL infini pour tous les écarts considérés puisque toutes les données sont identiques.

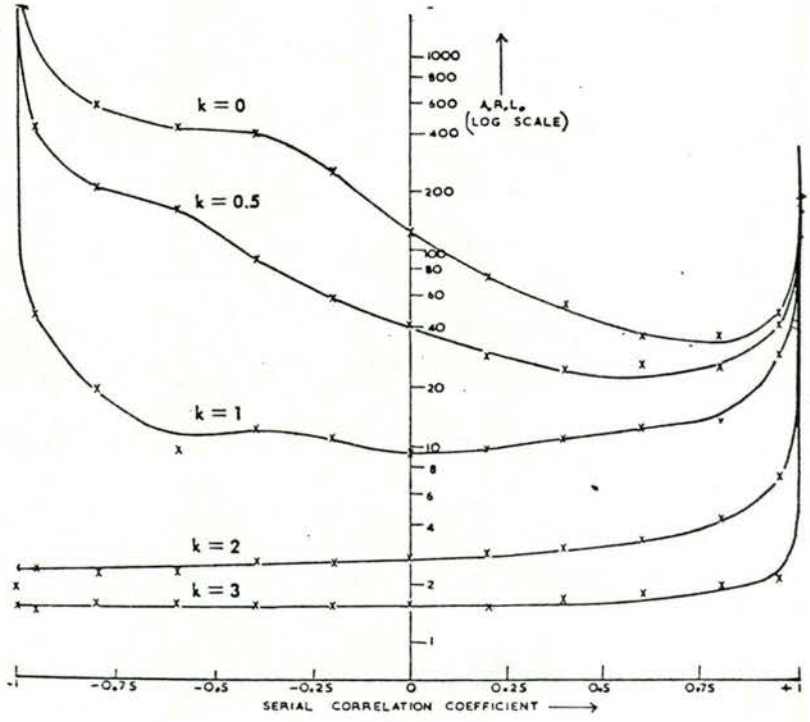
a = - 1

Le coefficient de corrélation est égal à $\pm$ 1.

$x_{2i-1} = k\sigma - y_0 \qquad x_{2i} = k\sigma + y_0$

$S_{2n-1} = (2n-1)\,k\sigma - y_0 \qquad S_{2n} = 2\,nk\sigma$

On aura donc un ARL infini quand la déviation recherchée par rapport à la moyenne sera inférieure ou égale à tg θ .
Pour les grandes déviations, l'ARL se calcule à partir des tables de distribution normale.
La figure ci-dessous nous donne les ARL ($k\sigma$) pour des k = 0, 0.5, 1, 2, 3 . Le V-écran a été construit avec d = 2 et tg θ = 0.5.



On voit que pour les grandes déviations par rapport à la moyenne, c'est-à-dire avec k = 2 ou 3, la corrélation sérielle a peu d'influence sur les longueurs moyennes de séries sauf au voisinage de corr () = 1.
Par contre, pour les petites valeurs de k, une corrélation positive augmente la rapidité de détection des écarts (et ainsi fait parfois diminuer les ARL d'un facteur 2) tandis qu'une corrélation négative va augmenter considérablement les ARL et ainsi retarder très fortement les détections d'erreurs.

Il est important de détecter une corrélation positive pour bien se fixer les paramètres d et θ afin d'éviter de déclencher trop souvent une interruption alors que la donnée devrait être considérée comme correcte.
De même pour une corrélation négative, cela mènerait à accepter des données incorrectes ou de les détecter avec un retard inacceptable.

II.4. Taille des échantillons.

Comme il est nécessaire pour tester les données de posséder une bonne estimation de la moyenne et de l'écart-type, la taille de l'échantillon se doit de n'être pas trop exiguë pour diminuer au maximum l'erreur due à l'estimation. On se propose donc de calculer le nombre n de données à envisager de telle sorte que l'erreur relative ne dépasse pas 10 % dans l'estimation de l'écart-type au niveau d'incertitude de 5 %.
On ne tiendra pas compte de l'estimation de la moyenne car celle-ci donne déjà de très bons résultats à partir de cinquante données.

Intervalle de confiance pour l'estimation de σ pour une variable normale.

Si on prend un nombre de données assez grand, plus de deux cents, l'intervalle de confiance peut s'écrire :

$$\Pr\left[\underline{s} - \frac{Q_g(1-\alpha'')\,\underline{s}}{\sqrt{2n}} \leqslant \sigma \leqslant \underline{s} - \frac{Q_g(\alpha')\,\underline{s}}{\sqrt{2n}}\right] = 1 - \alpha' - \alpha''$$

Soit $\alpha/2 = \alpha' = \alpha''$

$$\Pr\left[-Q_g(1-\alpha/2) \leqslant \frac{\sigma - \underline{s}}{\underline{s}/\sqrt{2n}} \leqslant Q_g(1-\alpha/2)\right] = 1 - \alpha/2 - \frac{\alpha}{2}$$

Soit $\alpha = 0.05$ niveau d'incertitude

$$\Pr\left[\frac{\sigma - \underline{s}}{\underline{s}/\sqrt{2n}} \leqslant Q_g(1-\alpha/2)\right] = 1 - \frac{0.05}{2} = 0.975$$

or on a $Q_g(0.975) = 1.96$

d'où $$\frac{\sigma - \underline{s}}{\underline{s}} \leqslant \frac{1.96}{\sqrt{2n}}$$

comme on a choisi $n \geqslant 200$

on a donc : $$\frac{\sigma - \underline{s}}{\underline{s}} \leqslant \frac{1.96}{\sqrt{2 \times 200}} = \frac{1.96}{20} = 0.098$$

L'erreur relative est de 9.8 %. On voulait une erreur plus petite que 10 %, l'hypothèse est donc vérifiée et l'on choisira pour nos évaluations futures des échantillons de taille plus grande ou égale à 200.

II.5. Critères de choix.

Avant de se préoccuper du problème pratique à résoudre, il faudrait peut-être se demander si les trois méthodes proposées ne donnent pas les mêmes résultats et surtout ne pourrait-on pas essayer d'évaluer leurs performances sur base de certains critères ?

Le critère auquel il aurait fallu se rattacher en premier lieu est sans conteste la fiabilité des méthodes.
Le problème est que c'est pratiquement impossible à mettre en oeuvre étant donné la subjectivité du problème à résoudre.
En effet, une donnée détectée fausse par une méthode peut être tout de même acceptée par l'utilisateur et inversément.
On pourrait alors marquer certaines données comme tout à fait incorrectes et tester si les méthodes les détectent.
Mais il suffira de choisir un intervalle de décision assez petit pour que chaque méthode fasse ressortir ces données ainsi que d'autres qui devraient être acceptées. On ne peut donc parler de fiabilité !

Quand on traitera des données pratiques, qui ne possèdent pas tout à fait une distribution normale, il sera peut-être alors possible de classifier les méthodes quant à leur fiabilité (d'après les remarques que l'on tirera sur des essais pratiques !).

Pour essayer de trouver un autre critère, on va tester les méthodes sur des populations théoriques normales.

Voir annexe II, pages 1 à 3 la manière dont la population normale a été obtenue et les données dont on disposera de ce fait.

On applique dès lors les trois méthodes sur la population théorique en calculant les paramètres tels que ARL (2σ) = 3.

Une première remarque est que les ARL (0) respectifs sont de 250, 150 et 380.

Comme on a forcé un écart de 2σ par rapport à la moyenne, on va pouvoir évaluer un ARL (2σ) en calculant la longueur moyenne des séries avant détection d'erreur.

Voici les résultats obtenus :

Méthode I : longueurs des séries = 4, 4, 2, 5, 3, 4, 2, 1, 7, 3, 4, 1, 3, 3

longueur moyenne = 3.29

écart-type des longueurs = 1.59

Méthode 2 : longueurs des séries = 4, 4, 2, 5, 3, 4, 3, 7, 3, 4, 2, 3, 3.

longueur moyenne = 3.62

écart-type des longueurs = 1.33

Méthode 3 : longueurs des séries = 3, 3, 3, 2, 4, 3, 4, 2, 1, 5, 2, 3, 3, 2, 3, 3

longueur moyenne = 2.88

écart-type des longueurs = 0.96

On voit tout de suite que la moyenne obtenue par la deuxième méthode est beaucoup plus élevée que celles obtenues par les deux autres. Cela entraîne une lenteur évidente à la prise de décision.

Par contre, les deux autres moyennes sont relativement proches de l'objectif qui était de 3 au départ (de 4 à 10 % d'erreur relative !). On peut maintenant les différencier si l'on considère l'écart-type obtenu en prenant les longueurs des séries comme données. Celui de la troisième méthode est en effet de 40 % plus petit que celui de la première méthode ! (les valeurs s'échelonnent de 1 à 5 au lieu de 1 à 7).

C'est dire que la troisième méthode semble moins sensible aux grandes déviations brusques mais brèves.
Comme c'est un des buts recherchés dans le cas qui nous intéresse, on peut donc faire ici une première classification des méthodes (surtout si on tient compte du fait que les ARL(0) obtenus font ressortir la même échelle des valeurs) : la troisième méthode semble la meilleure, elle serait suivie de la première puis de la seconde.

Il faut aussi remarquer que la simplicité de la troisième méthode renforce encore cette conclusion !

En effet, cette méthode ne fait varier qu'un paramètre (h) alors que les deux autres font varier parallèlement d et θ .

Une étude sur plus de données confirmerait ou non cette conclusion.

Quant à nous, on va se baser sur des données réelles pour essayer de mieux classifier les méthodes et d'en retirer les différences essentielles. On pourra toujours tenir compte de la remarque ci-dessus pour confirmer ou mettre en réserve nos futurs résultats.

CHAPITRE III

ADAPTATION DES METHODES

TESTS SUR DES POPULATIONS REELLES

- Calcul par variation d'une double somme
- Différentes méthodes + exemple commun aux trois méthodes
- Première conclusion
- Résultats de tests effectués sur des populations réelles de résultats d'analyses médicales
- Conclusions

III.1. Application pratique

Adaptation des méthodes pour l'implémentation

III.1.1. Calcul par variation d'une double somme.

On a vu au chapitre 1 que l'on a essentiellement trois méthodes à notre disposition pour effectuer un contrôle de qualité convenable :

1. V-écran de Ewan et Kemp
2. V-écran de Goldsmith et Whitfield
3. intervalle de décision

Ces trois méthodes diffèrent dans la manière d'estimer les paramètres d et θ du V-écran ou l'intervalle de décision. Pour les deux premières, les conditions de sortie de l'écran étaient :

$$\exists\ r \text{ tq } \sum_{i=n-r}^{n} (x_i - \mu - w \operatorname{tg}\theta) \geqslant d \operatorname{tg}\theta$$

c'est-à-dire une sortie par le bas ou encore une tendance croissante du processus par rapport à la valeur centrale de référence.

ou

$$\exists\ r \text{ t}_q \sum_{i=n-r}^{n} (x_i - \mu + w \operatorname{tg}\theta) \leqslant - d \operatorname{tg}\theta$$

c'est-à-dire sortie par le haut impliquée par un abaissement de la moyenne courante du processus par rapport à la valeur de référence .

On peut aisément tester ces deux conditions en considérant deux accumulateurs X et Y dont la mise à jour s'effectue comme suit :

$$\text{soit} \begin{cases} x_i' = x_i - \mu - w \operatorname{tg}\theta \\ x_i'' = x_i - \mu + w \operatorname{tg}\theta \\ X_0 = Y_0 = 0 \end{cases}$$

On a : $\begin{cases} X_i = \max (0, X_{i-1} + x_i') \\ Y_i = \min (0, Y_{i-1} + x_i'') \end{cases}$

La sortie du V-écran se produit quand :

$X_i \geqslant d \operatorname{tg} \theta$ ou $Y_i \leqslant - d \operatorname{tg} \theta$

Pour ce qui est de la troisième méthode, on peut se servir des accumulateurs mais en posant $w \operatorname{tg} \theta = 0$

$d \operatorname{tg} \theta$ = intervalle de décision = h.

III.1.2. Utilisation pratique de la méthode 1 de Ewan et Kemp :

On connaît la valeur cible μ et l'écart-type σ.
On choisit le pas w.
On choisit la déviation que l'on veut détecter le plus rapidement possible (souvent $\Delta\mu = 2\sigma$).
On a le choix alors de fixer une des variables parmi ARL(0), ARL ($\Delta\mu$), d et θ .
Le plus naturel est de choisir une valeur pas trop grande pour ARL ($\Delta\mu$) et de chercher les autres.

On peut ainsi calculer $\frac{1}{2} \frac{\Delta\mu}{w}$ qui d'après Ewan et Kemp sera égal à $\operatorname{tg} \theta$.

On cherche dans le tableau du paragraphe I.2.2. une valeur du rapport $\frac{1}{2} \frac{\Delta\mu}{\sigma}$ égale à celle qu'on a choisie (d'après le choix de $\Delta\mu$) et cela dans la ligne correspondant à l'ARL ($\Delta\mu$) que l'on s'est fixé comme objectif.
On trouve ainsi l'ARL (0) correspondant à la colonne où se situe le rapport $\frac{1}{2} \frac{\Delta\mu}{\sigma}$ trouvé.
De même, on a alors à notre disposition le rapport $\frac{d}{w}$.

Comme on a choisi le pas au départ, on peut donc calculer d.

Remarque : on a la relation $\operatorname{tg} \theta = \frac{1}{2} \frac{\Delta\mu}{w}$: c'est-à-dire que le θ que l'on trouve est fonction de $\Delta\mu$! ?

Exemple

soient : $\begin{cases} \mu = 100 \quad \sigma = 15 \quad w = 1.0 \quad \Delta\mu = 2\sigma \\ ARL\ (\Delta\mu) = 3 \end{cases}$

On obtient : $tg\ \theta = \frac{1}{2} \frac{\Delta\mu}{w} = 1.5$

$\frac{1}{2} \frac{\Delta\mu}{\sigma} = 1$ valeur du rapport approchant le plus est 1.03 correspondant à un ARL (0) = 250.

On a aussi $\frac{d}{w} = 2.20$ ou encore $d = 22.0$

On a donc : $\begin{cases} w\ tg\ \theta = 15 \\ d\ tg\ \theta = 33 \end{cases}$

Une analyse plus complète du tableau fait apparaître certaines restrictions quant à l'utilisation de cette méthode.
En effet, supposons que l'on veuille un ARL (2σ) égal à 6.

On aurait donc $\frac{1}{2} \frac{\Delta\mu}{\sigma} = 1$ mais dans la ligne correspondant à ARL (2σ) = 6, la valeur la plus proche est seulement 0.72.
On détecte donc en fait $\Delta\mu = 0.72 \times 2\sigma$ au lieu de 2σ .
Ce n'est pas ce que l'on voulait au départ !

On n'a également pas beaucoup de choix si l'on désire étudier plusieurs possibilités quant aux variables d, θ et ARL (0).
En fait, il semble que ce tableau ne s'utilise pas de cette manière mais bien en se fixant au départ les ARL (0) et ARL ($\Delta\mu$) que l'on veut pouvoir détecter.
On pourrait ainsi déterminer le $\Delta\mu$, la distance d et l'ouverture θ .

III.1.3 Utilisation pratique de la méthode 2 de Goldsmith & Whitfield

On a à notre disposition un ensemble de courbes (pages Ann. I.7 & 8) résultant de simulation.
On peut déjà remarquer que θ ne dépend plus du $\Delta\mu$ choisi.
Il suffira au départ de se choisir le $\Delta\mu$ que l'on veut détecter et l'ARL ($\Delta\mu$) désiré.

Le choix de la courbe nous fournira alors d et θ. On pourra aussi trouver aisément l'ARL(0).

exemple

si $\Delta\mu = 2\sigma$ ARL(2σ) = 5 on a le choix entre les courbes IX, XIII, XVI.

IX	d = 2	tg θ = .7	ARL(0) $\gg$ 1500
XIII	d = 5	tg θ = .45	ARL(0) $\gg$ 2000
XVI	d = 8	tg θ = .35	ARL(0) $\gg$ 1500

Quelle courbe choisir ?

Toutes trois proposent des ARL(0) amplement suffisants ; on ne peut donc tenir compte de ce critère !

Considérons par contre les ARL(σ) : il vient respectivement 80, 28 et 19. Il est donc probable que la courbe XVI soit la meilleure.

En fait, il semble intéressant d'avoir d grand et θ petit ; si on examine de plus près l'expression qui détermine la sortie du V-écran, c'est-à-dire

$$\sum_i (x_i - \mu \pm w \operatorname{tg}\theta) \lessgtr \mp d \operatorname{tg}\theta \ ,$$

$w \operatorname{tg}\theta$ étant en quelque sorte un écart rectificateur, les résultats seront plus réguliers si l'on choisit un θ petit et d grand. En effet, on aura $d \operatorname{tg}\theta$ qui ne variera pas beaucoup tandis que $w \operatorname{tg}\theta$ diminuera avec θ . On revient ici au critère, envisagé au paragraphe II.5, qui était la variance du nombre de générations avant signal correctif. Cette variance sera d'autant plus minimale que l'expression sera régulière ou encore que d sera grand et θ petit.

Application pratique

Comme les abscisses des diagrammes sont graduées en terme de 2σ, il est nécessaire de modifier les valeurs proposées de d et $\operatorname{tg}\theta$ dans les tableaux.

$$d_{réel} = d_{lu} \times \sigma$$

$$\operatorname{tg}\theta_{réel} = \operatorname{tg}\theta_{lu}/\alpha \quad \text{où } \alpha = \frac{w}{2\sigma}$$

Exemple

soient $\mu = 100$ $\sigma = 15$ $\Delta\mu = 2\sigma$ $w = 10$

$ARL(2\sigma) = 3$

on choisit la courbe III

on lit dans le tableau : $\begin{cases} d = 1 \\ \operatorname{tg}\theta = 0.7 \end{cases}$

on trouve un $ARL(0) = 150$

on aura donc $\begin{cases} d = 15 \\ \operatorname{tg}\theta = 2.1 \end{cases}$

on a finalement : $\begin{cases} w \operatorname{tg}\theta = 21 \\ d \operatorname{tg}\theta = 31.5 \end{cases}$

III.1.4 Utilisation pratique de la méthode 3 basée sur l'intervalle de décision.

Pour cette méthode, on n'utilise plus de V-écran. On n'aura plus qu'une seule variable h à évaluer pour effectuer le contrôle de qualité.

Les accumulateurs seront construits comme suit :

$$\sum_{i=n-r}^{n} (x_i - \mu)$$

La sortie vers le haut se produira quand $Y_i \leqslant -h$

La sortie vers le bas se produira quand $X_i \geqslant h$

Ici encore, le h lu sur les abaques A et B du paragraphe I.2.5 doit être modifié en raison de la graduation en 2σ.
En fait, le h que l'on lit est égal à $h/2\sigma$.

$$\text{d'où} = h_{réel} = h_{lu} \times 2\sigma$$

Exemple

Soient toujours $\begin{cases} \mu = 100 \quad \sigma = 15 \quad w = 0 \quad \Delta\mu = 2\sigma = 30 \\ ARL(2\sigma) = 3 \end{cases}$

$\frac{\Delta\mu}{2\sigma} = 1$, on reporte cette valeur à droite sur l'abaque A

ARL $(\Delta\mu)=3$ on reporte 3 à gauche sur l'abaque A

on trouve ainsi $h_{lu} = 2.2$

On porte de nouveau $\frac{\Delta\mu}{2\sigma} = 1$ à gauche sur l'abaque B

La droite joignant les deux points coupe la verticale donnant les ARL(0) au point 380 .

on obtient donc un h = 2.2 x 2 x 15 = 66

Rem : cette valeur est sensiblement plus grande que dans les deux premiers cas (33 et 31.5) . Ceci est dû au fait que w = 0.

Conclusion théorique.

Par l'exemple que l'on a traité avec les trois méthodes, si l'on tient compte des ARL(0) obtenus :

1) 250
2) 150
3) 380

il semble que la troisième méthode soit de loin préférable aux deux autres.
Il faut remarquer que c'est déjà la conclusion qui avait été tirée après l'analyse statistique du chapitre II.

Il nous reste maintenant à tester ces méthodes sur des populations de résultats réels d'analyses médicales pour voir si la même tendance se généralise.

Si c'est le cas, on pourra en tirer des conclusions assez strictes et fiables. Sinon, il faudra émettre des réserves quant à l'utilisation d'une méthode plutôt qu'une autre.

III.2 Essais des méthodes sur données réelles

On trouvera à l'annexe III p. 1 à 4 le programme ayant permis de faire ces quelques essais préliminaires !

III.2.1. Lipides totaux normaux. (voir annexe III. p. 5 et 6)

Pas de sortie pour la méthode 1 : trop peu sensible.

Méthode 2 : sortie unique après 27 coups = normal, mais laisse passer une augmentation dans la moyenne courante du processus entre la cinquième et la huitième donnée (590, 590, 610, 590).

Méthode 3 : peut-être trop sensible mais détecte, elle, l'augmentation décrite ci-dessus après 10 coups ; elle réagit en plus à une baisse qui se manifeste après la donnée 18 (570, 580, 590, 570, 580, 570, 570) ⟶ sortie après 22 coups. Elle trouve aussi la sortie après 27 coups comme la méthode 2 l'avait fait.
- La suite est beaucoup plus aléatoire et moins fiable.

Puisqu'on estime la méthode trois trop sensible, essayons-la en choisissant un ARL(2σ) = 4.
La tendance à la hausse aux environs de la septième donnée est passée inaperçue ! Par contre, la baisse détectée auparavant au 27^{è} coup l'est ici au 23^{è} . Celle détectée un peu à la légère au 39^{è} l'est ici au 35^{è} et se justifie beaucoup mieux (succession de 580 plus un 570) !

III.2.2 Lipides totaux pathologiques. (voir annexe III p. 6 et 7)

Les deux premières méthodes donnent des résultats presque

pareils, la deuxième étant un peu plus rapide.

Mais de nouveau, la troisième méthode est beaucoup trop sensible. Les erreurs détectées par les deux premières le sont aussi par la troisième mais il y a d'autres sorties peu intéressantes.

Si l'on choisit un ARL (2σ) = 4, on a moins de sorties mais les principales ne sont pas oubliées et sont même détectées plus rapidement.

Rem : Il y a un choix à faire semble-t-il entre la rapidité de détection des erreurs et la confiance que l'on peut avoir en cas de sortie.

III.2.3 Bilirubine totale normale. (Voir annexe III. p. 8 et 9)

Ici, la méthode I est de loin trop peu sensible en détectant deux erreurs seulement.

La méthode 2 donne des résultats tout à fait farfelus en signalant beaucoup trop de bonnes données comme étant incorrectes.

De nouveau, la méthode 3 semble un peu trop sensible.

Mais dans ce cas, si l'on choisit un ARL(2σ) = 4 , cette même méthode donne des résultats acceptables et certainement beaucoup plus fiables.

III.2.4 Bilirubine totale pathologique. (Voir annexe III 10 et 11)

On a cette fois des résultats semblables à ceux obtenus pour la bilirubine totale normale (excepté le fait que la seconde méthode réagit mieux en détectant seulement 4 erreurs). Mais ici encore, c'est la méthode 3

avec ARL(2σ) = 4 qui donne les meilleurs résultats.

III.2.5 Analyse plus approfondie

Pourquoi avoir choisi ces deux types de sérums ? Ils se différencient principalement par le fait que l'un (lipides totaux) présente des coefficients de variation très faibles (1.69 et 1.90 %) tandis que l'autre (bilirubine totale) en possède de très élevés (20.23 et 15.62 %).

Au point de vue résultats des méthodes, pour la bilirubine, c'est la méthode trois qui l'emporte de loin (avec ARL (2σ) = 4) tandis que pour les lipides, ce serait plutôt la seconde ou à la rigueur la troisième à condition d'augmenter l'ARL(2σ) choisi.

Voyons si le critère du coefficient de variation peut se révéler efficient. D'autres essais sur chlorure normal et pathologique (coeff. de variation très faibles) mettent en évidence la deuxième méthode tandis que ceux effectués sur acide urique normal, lithium normal et pathologique (coeff. très élevés) font ressortir la troisième méthode (avec pourtant ARL(2σ) = 3).
On peut donc considérer le coefficient de variation de la population à étudier comme critère de choix des différentes méthodes :

coeff. élevé	méthode 3
coeff. faible	méthode 2

La méthode 1 quant à elle semble beaucoup trop lente à réagir et comme ARL(2σ) = 3 est la plus petite valeur à notre disposition et que l'on a peu de possibilités de variations des paramètres (cfr paragraphe III.1.2.), elle paraît bien faible à côté des deux autres méthodes !

III.3 Conclusions

Après avoir éliminé définitivement la première méthode, on peut ainsi résumer les principales conclusions quant à l'utilisation de la deuxième ou de la troisième méthode.

Favorable à la seconde.

- en cas de population présentant un coefficient de variation assez bas, détecte les erreurs importantes sans en détecter d'autres plus futiles.

Favorables à la troisième.

- beaucoup plus simple pour essayer plusieurs solutions du fait que l'on n'a qu'un facteur à faire varier (h).
- variance des résultats sur population théorique plus faible.
- meilleure ARL(0), beaucoup plus grande..
- propose toujours une solution quand le coefficient de variation est élevé.
- solution acceptable quand le coefficient de variation est assez bas. (Par acceptable, on entend qu'une erreur ne soit pas oubliée par le contrôle de qualité même s'il en détecte qui ne sont pas tout à fait incorrectes.)

Ces conclusions vont être très importantes dans l'application de l'interactif au contrôle de qualité, décrite dans le chapitre suivant.

CHAPITRE IV

Utilisation en interactif.

-principe.
-organisation des fichiers.
-création des fichiers.
-initialisation des fichiers.
-détermination des paramètres.
-ajout d'une donnée .
-recherche de l'intervalle de décision optimal.
-Langage de programmation utilisé.
-explications et fonctionnement des programmes . creat
. init
. ajout
. qc

IV.1 Utilisation en interactif

Pour que le contrôle de qualité remplisse son rôle premier de détection rapide des déviations des résultats, il ne fait aucun doute que chaque donnée doive être traitée immédiatement après son apparition. L'arrivée d'une donnée déclenchera donc une phase de stockage de cette donnée dans un fichier suivie d'un contrôle de qualité sur cette donnée par rapport à celles qui l'ont précédée.

Le but que l'on s'est fixé au départ est que l'utilisateur effectue son contrôle de qualité sans nullement se préoccuper de recherche de paramètres, variances, moyennes ou autres variables. Il ne doit donc exister un dialogue entre lui et la machine qu'en termes intelligibles pour les deux parties (et surtout pour l'utilisateur). On exclut donc la possibilité que l'utilisateur effectue le moindre calcul ; ce sera à la machine de lui proposer des solutions compréhensibles qu'il lui sera libre d'accepter ou de refuser.

IV.1.1. Principe

Pour les données passées, l'utilisateur peut les qualifier correctes ou incorrectes selon les observations qu'il a pu relever en pratique. Donc, si l'on dispose d'un certain nombre de données en plus du fait que l'on sait si elles sont correctes ou non, on peut préparer le contrôle de qualité de la prochaine donnée en faisant une recherche initiale.

Cette recherche consiste à simuler un contrôle de qualité sur les anciennes données et de conserver les paramètres qui ont donné le meilleur résultat. Ce résultat est obtenu en faisant varier les paramètres nécessaires au con-

trôle de qualité et il est tel que toutes les erreurs que l'utilisateur a détectées en pratique le soient aussi par le programme.
Une fois ce résultat trouvé, il s'agit de stocker en mémoire les paramètres correspondants pour ne pas devoir les recalculer à chaque introduction de nouvelle donnée.

C'est ici que le dialogue interactif va trouver son plein développement car il est peu probable que sur un grand nombre de données, le programme détecte exactement toutes celles voulues incorrectes : il peut en effet en oublier, en détecter d'autres à tort, en détecter des vraies mais un coup ou plusieurs en avance ou en retard, par rapport à l'estimation de l'utilisateur.
Il doit donc proposer des solutions à l'utilisateur qui est libre à son tour de les accepter ou de les refuser. S'il en accepte une, il y a copie en début de fichier des paramètres ayant fourni cette solution ; sinon, il y a recherche d'une autre solution tant que cela est possible et peut être utile. Si l'utilisateur refuse toute solution, il s'agit pour lui de contrôler la vraisemblance de ses affirmations quant à la justesse ou non des données passées.
S'il peut en corriger, il le fait. Sinon, il faut qu'il prenne un risque en choisissant des paramètres tels qu'il n'oublie pas des données incorrectes mais qu'il en détecte d'autres à tort. En fait, le fait de refuser toute solution implique le forçage des paramètres à des valeurs telles que la longueur moyenne des séries quand la moyenne courante s'écarte de 2σ de la moyenne est égale à 3 (c'est-à-dire ARL $(2\sigma) = 3$).
En effet, les conclusions du chapitre III nous indiquent que toutes les données fausses seront détectées mais qu'il y en aura d'autres qui le seront à tort.

La probabilité d'erreur est donc plus grande mais ça ne représente pas une erreur grave car il est moins grave de détecter une erreur en trop que d'en oublier une et de ce fait de considérer comme correcte une donnée qui ne l'est point.

IV.I.2. Application pratique

Comme on a déduit (chap. II et III) que la méthode 3 donnait les meilleurs résultats et que de toute manière ceux produits par la deuxième y étaient inclus, on va uniquement travailler avec la troisième.

Rappelons que cette méthode est basée sur l'intervalle de décision h tel que l'on détecte une erreur si

$$\Sigma (x_i - \mu) \geq h$$

ou puisque w = 0

$$\Sigma (x_i - \mu) \leq -h$$

On n'a donc que le paramètre h à faire varier de sa plus petite valeur à sa plus grande pour trouver toutes les solutions possibles à notre problème.

Le h à considérer est égal à : $h_{lu} \times 2 \times \sigma$ où le h_{lu} est égal à une valeur trouvée sur les abaques A et B du chapitre I. On peut ainsi y trouver un h_{lu} pouvant varier de 2 à 5. On va donc commencer la simulation avec un $h_{lu} = 2$ et l'incrémenter de 0.1 à chaque itération jusqu'à ce qu'il atteigne 5 au maximum si cela s'avère encore utile.

IV.I.3 Organisation des fichiers

On dispose d'un fichier par type de données.

Ces fichiers sont organisés de la manière suivante :

1) paramètres
2) données.

Les paramètres occupent une zone fixe en début de fichier. Ils se répartissent en dix nombres réels (en floating point) de 4 bytes chacun. Cette zone occupe donc 40 bytes. On décrira plus tard chaque réel en particulier mais sachons déjà qu'il y aura là tout ce qui sera nécessaire pour effectuer n'importe quel contrôle de qualité sur les données qui suivent.

Chaque donnée sera composée d'un réel qui la représente plus un caractère (1 byte) disant si elle est correcte ou non. Ce sera donc un groupe de 5 bytes par donnée.

Rem. En fait, comme il y a alignement des nombres en floating point sur les mots et que les mots prennent deux bytes, il y aura chaque fois une perte d'un byte entre chaque groupe composé d'une donnée et d'un caractère.

IV.I.4 Création des fichiers

Avant de travailler sur des données et de les stocker dans un fichier, il faut évidemment créer le fichier. On le fait au moyen du programme creat qui crée un fichier au nom qu'on lui indique, fichier accessible en lecture et écriture par tout utilisateur.

IV.1.5 Initialisation des fichiers.

Comme un bon contrôle de qualité requiert un minimum de deux cents données pour être statistiquement vala-

ble, il faut avant toute chose procéder à une initialisation du fichier dans laquelle il est nécessaire de stocker les deux cents données initiales ainsi que les paramètres qu'il est déjà possible de calculer (somme des données, somme des carrés, nombre de bytes écrits sur le fichier, format d'impression des données).
Chaque donnée se doit évidemment d'être accompagnée de son caractère de justesse : 1 ou 0 selon que la donnée est considérée comme correcte ou incorrecte.
Cette initialisation sera faite par la programme init.

IV.1.6 Détermination des paramètres.

Une fois l'initialisation effectuée, il faut préparer le contrôle de qualité de la prochaine donnée grâce aux renseignements que l'on possède sur les deux cents premières données. Il s'agit donc de compléter la liste de paramètres de début de fichier dont certains éléments ont déjà été calculés par init. Cette initialisation de paramètres va se faire par le programme qc.
La liste de paramètres se compose comme suit :

1) réel = 1 si on a plus de 210 données dans le fichier
0 sinon

Ce nombre est nécessaire car on veut se limiter aux 200 dernières données, pour avoir une situation aussi proche que possible de la réalité actuelle (en cas de changement dans la moyenne des processus). Quand on en aura plus dans le fichier, on devra les retrouver à partir de la fin du fichier tandis que si l'on en a moins, on pourra les lire directement après le quarantième byte et cela jusqu'à la fin du fichier.

2) somme des données qui nous intéressent

3) somme des carrés des données qui nous intéressent. On a besoin de ces deux sommes car pour effectuer le contrôle de qualité, on a besoin d'une estimation de la moyenne et de l'écart-type.

4) nombre de bytes occupés par les données qui nous intéressent. Il a semblé préférable de mémoriser le nombre de bytes plutôt que le nombre de données car les recherches dans les fichiers s'effectuent en nombre de bytes. Si bien que l'on possède déjà sans cal-

cul un moyen de se positionner à l'endroit désiré.

5) format d'impression des données. On dispose de deux types de données : entiers et réels. Pour la beauté de l'impression des résultats, on voudrait éviter des zéros non significatifs pour les entiers ainsi que pour les réels (les données dont on dispose se limitent à deux chiffres significatifs après la virgule).

On aura donc une impression :

pour les entiers ----------%f0
pour les réels ----------%f2

6) écart = w tgθ . Le programme ayant été prévu pour utiliser n'importe quelle méthode de contrôle de qualité, on est obligé de mémoriser w tgθ qui intervient dans les deux premières méthodes. Comme on utilise la troisième, il suffit de mettre ce paramètre à zéro dans init pour rester cohérent.

7) premier accumulateur

8) deuxième accumulateur.
Pour préparer le contrôle de la prochaine donnée à entrer dans le fichier, il est nécessaire de conserver l'état des deux accumulateurs puisque ce sont eux qui déterminent les conditions de sortie.

9) intervalle de décision : c'est le but premier du programme qc. Une fois qu'il est déterminé après l'initialisation et un premier passage de qc, c'est lui qui servira au contrôle de qualité des données futures tant que cela satisfera l'utilisateur. Une fois

que les solutions du contrôle de qualité ne coïncideront plus avec les exigences de l'utilisateur, il faudra faire repasser le programme qc pour déterminer un nouvel intervalle de décision.

10) moyenne. C'est celle qui a servi à déterminer le h ci-dessus et qui servira au contrôle des prochaines données. On ne peut pas se permettre de recalculer la moyenne à chaque ajout de donnée car le fait d'ajouter une donnée va modifier les sommes 2) et 3) ainsi que le nombre de bytes 4). La moyenne changerait donc ainsi que l'écart-type et on effectuerait un contrôle avec des paramètres ne correspondant plus à ceux estimés pour les données antérieures.

Rem. 1 : Les 6 premiers paramètres seront initialisés par le programme init tandis que les 4 derniers le seront par qc.

Rem. 2 : On remarque que certains paramètres ne nécessitaient pas l'emploi d'un nombre en floating point mais qu'un seul caractère aurait suffi (paramètres 1,5) ou un entier (paramètre 4).
On a choisi de les mettre tous en floating point du fait que la perte de place n'était pas importante et surtout pour pouvoir travailler à l'aide d'une matrice et ainsi accéder à tous les paramètres à l'aide d'une seule opération de lecture/écriture.

IV.1.7 Ajout d'une donnée.

Après une lecture des paramètres de début de fichier, on a tout ce qui est nécessaire pour effectuer le contrôle de qualité de la donnée à ajouter au fichier. Une fois qu'elle aura été étiquetée correcte (1) ou incorrecte (0) et acceptée par l'utilisateur, elle est recopiée avec son caractère en fin de fichier.
Il faut aussi ajuster les paramètres sommes, nombre de bytes et accumulateurs.
Enfin, si le nombre de données devient plus grand que 210, il sera nécessaire de le diminuer. Pour cela un traitement est obligatoire car un contrôle de qualité ne peut pas démarrer dans n'importe quelles conditions : il faut que les accumulateurs soient initialisés à zéro. Or, on ne les remet à zéro qu'après une donnée fausse. Si bien que il faudra détecter la première donnée fausse et se positionner juste après pour obtenir la première donnée qui nous intéressera dans le futur (dans le cas où il faudra rechercher un nouvel intervalle de décision).

IV.1.8 Recherche de l'intervalle de décision optimal.

Tout d'abord, il est nécessaire de lire les cinq premiers paramètres. On peut ainsi calculer la moyenne et l'écart-type, et de plus on connaît le nombre et la position des données qui nous intéressent.
Avec l'intervalle de décision minimum, on peut effectuer le premier contrôle de qualité sur toutes les données que l'on a lues. Cela fournit une solution qu'il faut maintenant tester : cette solution se présente sous la forme d'un vecteur binaire dont le nombre de composantes

est égal au nombre de données considérées. Il s'agit donc de comparer chaque caractère associé à une donnée et lu dans le fichier avec la composante correspondante du vecteur résultat.
Si tous les caractères coïncident parfaitement, on a obtenu une solution exacte.
Sinon, deux cas peuvent se produire : une donnée incorrecte n'est pas détectée par le programme ou une donnée correcte est détectée à tort par le programme. On considérera le premier cas comme étant une erreur grave par rapport au second car il est plus grave d'oublier une erreur que d'en détecter une en trop (car toute détection dénote tout de même d'un certain écart par rapport à la moyenne).
Il y a un cas intermédiaire dans lequel toutes les erreurs sont détectées mais imparfaitement. C'est-à-dire que l'on suppose au départ que ce que l'utilisateur a stocké dans le fichier est la limite qu'il ne faut pas dépasser ; et, dans ce cas, si une erreur est détectée un coup trop tôt, on ne peut que l'accepter ou du moins proposer la solution à l'utilisateur comme solution acceptable. Il a quant à lui la liberté totale de l'accepter ou de la refuser. S'il la refuse, elle est définitivement perdue.
Si la solution n'est ni exacte ni acceptable, elle est stockée selon le principe du minimum d'erreurs : c'est dire qu'à la fin de la simulation sera proposée la solution qui contiendra le moins d'erreurs (si possible sans erreurs graves, sinon avec !).
On atteindra la fin de la simulation si l'intervalle de décision servant au contrôle est si grand qu'on ne détecte plus aucune erreur ou qu'il dépasse un certain maximum.
Il va sans dire que le fait d'accepter une solution fait lui aussi finir la simulation après avoir recopié en début de fichier tous les paramètres correspondant à la solution retenue.

On a alors le choix de laisser les données telles quelles dans le fichier ou de modifier leur caractère de justesse d'après la solution acceptée. Il faudrait pour cela voir si cela va influencer fortement le contrôle de qualité suivant. Dans le cas où l'on veut recalculer l'intervalle de décision à cause d'une valeur qui est manifestement incorrecte mais cependant pas détectée par le programme, si auparavant on a recopié la solution acceptée (qui n'était pas exactement celle demandée au départ par l'utilisateur), il est pratiquement certain que le programme va proposer le même h et de ce fait la même solution que celle que l'on refuse maintenant. On sera ainsi dans une impasse et il faudra modifier assez bien de caractères pour avoir une chance de retrouver une solution acceptable. De plus, si chaque fois que l'on recalcule le h et que l'on trouve une solution approchée que l'on accepte, on la recopie dans le fichier, on risque de s'éloigner fortement du but initial que l'on s'était fixé et que l'on ne connaîtra même plus après la première acceptation. Le mieux est donc de conserver ce que l'on désire au départ pour qu'à chaque essai, on tende le plus possible vers cette solution idéale.

IV.2 Langage de programmation utilisé

Les programmes étant destinés à fonctionner sur PDP 11/45 ou 11/70 sous l'operating system UNIX, il a donc fallu les rédiger dans un langage accepté par cet OS. Il en étaient deux qui pouvaient convenir : le langage C et le fortran.

J'ai choisi au départ le C pour plusieurs raisons :

1) permet une programmation structurée aisée (sans goto).
2) Utilisation de pointeurs, de caractères et facilités procurées par l'emploi des structures pour condenser la description de grands ensembles de données.
3) Compilateur performant. Une bonne partie de l'OS lui-même est écrit en C.
4) Routines d'entrées/sorties relativement aisées d'utilisation.

Les entrées/sorties se faisant en ASCII, caractère par caractère, quand on entre une donnée au terminal, il faut lui appliquer un traitement pour la transformer en flottant ou en réel (routines atof et atoi).

Pour imprimer les données sur la sortie standard, on dispose de la routine printf qui permet d'imprimer tout ce que l'on désire selon le format choisi.

Etant donné que les données n'ont pas toutes le même nombre de chiffres, il est impossible de les stocker telles quelles dans un fichier si l'on veut les retrouver par la

suite. Il sera donc nécessaire de les traduire (ASCII to float) non plus avant toute opération arithmétique incluant l'emploi de cette donnée mais avant même le stockage de la donnée en mémoire ; grâce à cette opération, toutes les données en mémoire auront la même longueur de 4 bytes et il sera aisé de les retrouver toutes.

IV.3 Explications et fonctionnement des programmes.

IV.3.1 CREAT

voir annexe IV.1

Création d'un fichier dont on donne le nom. Ce nom peut avoir de I à 9 caractères.
Le fichier est créé avec le mode 0777 c'est-à-dire accessible en lecture/écriture par tout utilisateur.

IV.3.2 INIT

Voir annexe IV.2 à 4

Initialisation d'un fichier avant d'y effectuer tout contrôle de qualité.
Introduire le nom du fichier ; si le nom est incorrect, donc inconnu de la machine, il y a impression d'un message d'erreur suivi de la fin forcée du programme. Si le nom est correct, il est possible d'introduire au maximum 200 groupes (donnée, caractère).

Une fois toutes les données dans une zone tampon, on peut les écrire sur le fichier à partir du quarantième byte.
Il y a entretemps calcul de la somme et de la somme des carrés des données puis rangement dans les deuxième et troisième paramètres. Le premier paramètre est mis à zéro et le quatrième est fixé au nombre de bytes qu'occupent les données (nombre de données x 6).
Si les données sont entières, mettre le cinquième paramètre à zéro, sinon à un.

Recopier les paramètres ainsi trouvés en début de fichier.

Rem : Si on a plus de deux cents données, on ne tient compte que des deux cents dernières ; si on en a moins, on fait avec ce qué l'on a !

IV.3.3. AJOUT voir annexe IV.5 à 10

Ajouter une donnée au fichier en calculant son caractère de justesse selon les paramètres de début de fichier trouvés par le programme QC.
Introduire le nom du fichier (cfr ci-dessus).
Lecture des paramètres.
(a) Introduire la donnée à ajouter.
Ajustement des paramètres 2 à 4.
Contrôle de la qualité de la donnée.
Acceptation ou non par l'utilisateur du résultat fourni.
Copie de la donnée et de son caractère en fin de fichier.
Ensuite, il est nécessaire de tester le nombre de données puisque l'on veut toujours se limiter à plus ou moins 200 données (si l'on en prend plus, il deviendra impossible de visualiser clairement les solutions proposées).
Si le nombre de données dépasse 210, il faudra le restreindre, supprimer les premières données selon le principe expliqué précédemment qui veut que les accumulateurs doivent être nuls au départ d'un contrôle de qualité.
Une fois cette recherche faite, il faut modifier les paramètres affectés par cette opération.
Si l'on possède une autre donnée à ajouter au fichier, on recommence en (a), sinon on peut recopier les paramètres en début de fichier.

IV.3.4. QC voir annexe IV. 11 à 22

Détermination de l'intervalle de décision approprié aux 200 dernières données du fichier.
Introduction du nom du fichier.
Lecture des 5 premiers paramètres.
Lecture des 200 dernières données environ.
Calcul du nombre de données, de la moyenne, de l'écart-type.

Initialisation des variables.
Qualcon()
Stop.

Qualcon()

Incrémenter le h_{lu}

Calcul du h.

Détermination de la solution pour les données dont on dispose.

Test()

Test()

Test de la solution trouvée par qualcon()

Rem. On trouvera un organigramme décrivant en détail le test effectué pour évaluer la validité de la solution.

On peut obtenir :

solution exacte ... on peut stopper !
solution acceptable proposée : si acceptée stop
sinon qualcon()
solution finale : avec erreurs graves ou non
si acceptée stop
sinon révision des données.

Chaque fois qu'une solution est acceptée par l'utilisateur, il y a exécution de savepar() et copypar(). Il s'agit du sauvetage et de la copie des paramètres.
Si la solution n'est ni "exact" ni "proposed", il y a sauvetage selon le principe du minimum d'erreurs, ce qui peut entraîner l'exécution de savepar().
La fin de la simulation est atteinte quand le vecteur résultat de qualcon() est composé uniquement de 1 (il est

donc devenu inutile d'incrémenter encore h) ou quand le h_{lu} devient plus grand que 5.

Cette fin provoque l'impression de la meilleure solution obtenue selon le principe du minimum d'erreurs tout au long de la simulation.
Si cette solution est acceptée, il y a recopie des paramètres (copypar()).
Si cette dernière solution n'est pas acceptée, le h est forcé à 2.2 * 2 * σ, valeur correspondant à un ARL (2σ) = 3 pour une variable normale. Cet ordre de grandeur nous garantit en effet une détection rapide de futures erreurs. Il sera alors possible après l'introduction de quelques nouvelles données de recalculer un nouveau h dont les chances de découverte ne peuvent que croître !

CONCLUSION

Conclusion.

Il ressort de cette étude que l'utilisation des sommes cumulées permet un contrôle de qualité fiable et relativement aisé.

La comparaison effectuée entre les différentes méthodes de contrôle utilisant des diagrammes de sommes cumulées ou simplement des sommes cumulées fait apparaître que nul n'est besoin de quelque diagramme que ce soit.

Si le principe du contrôle par V-écran peut sembler intéressant pour visualiser au mieux les déviations de la moyenne courante, il semble par contre que l'utilisation est assez malaisée et si l'on veut travailler sur le diagramme, cela nécessite des manipulations extérieures à toute rationnalité (V-écran sur transparent superposé au diagramme où l'on porte successivement toutes les données).

Quelle que soit la manière dont sont estimés les paramètres du V-écran, cela demande tout de même un minimum de calculs et surtout un choix au départ (on devra ainsi estimer ARL(2σ), w, quelques fois ARL(0),...). On a vu que l'on disposait de deux méthodes d'estimation des paramètres d et θ du V-écran (Ewan & Kemp, Goldsmith & Withfield). La première s'est révélée trop restrictive quant aux possibilités de variation des paramètres et aussi d'exactitude des résultats obtenus.
La deuxième réclame un choix de courbe faisant varier les deux paramètres d et θ en sens inverse. Il est donc impossible de classifier ces courbes selon un ordre bien strict au point de vue des résultats obtenus.
C'est dire que si l'on veut utiliser cette méthode pour trouver les paramètres optimaux, il faudra essayer un

grand nombre de courbes avant de découvrir celle qui nous intéresse. C'est un peu laborieux surtout qu'il sera nécessaire en plus d'estimer w !

Etant donné cette difficulté en plus du fait que le but initial de cette étude était de proposer un contrôle de qualité ne réclamant aucun calcul de quelque difficulté que ce soit et devant fonctionner en interactif, il s'est vite révélé que la troisième méthode basée sur l'intervalle de décision présentait les meilleurs atouts en vue d'une implémentation en machine.

Le fait que l'on n'a plus qu'un paramètre à faire varier implique en effet qu'il est possible, en partant de la plus petite valeur jusqu'à la plus grande, de classifier tous les résultats obtenus sur base du nombre décroissant d'erreurs détectées. Il est ainsi aisément possible d'isoler la meilleure solution et le nombre de simulations à effectuer pour y arriver se trouve fortement réduit.

C'est donc cette méthode qui a été développée et implémentée mais tout a été prévu de telle sorte que si l'on veut implémenter une des deux autres, les changements ne soient pas trop importants.

Les essais effectués jusqu'à présent montrent toutefois que la troisième méthode satisfait amplement les desiderata des utilisateurs.

Ces essais prouvent aussi que de grands moyens de traitement ne sont pas nécessaires à l'implémentation d'un contrôle de qualité efficace (ici un PDP-11/45) .

On aurait pu pousser plus loin l'étude théorique des différentes méthodes et cela surtout au point de vue statistique. Là n'était pas notre but.
Il fallait avant tout un contrôle de qualité qui "marche" en interactif en laissant toute possibilité à l'utilisateur qui lui ne pouvait effectuer aucun calcul de quelque nature que ce soit.

C'est ce qui a été réalisé grâce à l'aide du laboratoire d'analyses médicales du Docteur M. Noel qui a mis à notre disposition son ordinateur et qui nous a permis de consulter à loisir les données qui nous étaient nécessaires. Nous l'en remercions vivement ainsi que toutes les personnes qui ont contribué de quelque manière que ce soit à la réalisation de ce mémoire.

BIBLIOGRAPHIE

REFERENCES

1) BARNARD G.A. "Cumulative Charts and Stochastic Processes"
Journal of the Royal Statistical Society - Série B - 1959 - 21 - 2, pages 239 / 271

2) BISSEL A.F. " Cusum Technics for Quality Control"
Applied Statistics, 18 (1-30), 1969

3) EWAN W.D. "When and How To Use Cusum Charts ?"
Technometrics 5, 1-22 (1963)

4)EWAN W.D. & KEMP K.W. "Sampling Inspection of Continuous Processes with no Autocorrelation Between Successive Results"
Biometrika,vol.47(1960),pp 363-380

5) FREUND R.A. "Graphical Process Control"
Industrial Quality Control, 1962, XVIII - 7, p. 15 / 22

6) GOLDSMITH P.L. & WHITFIELD H. "Average Run Lengths in Cumulative Chart Quality Control Schemes"
Technometrics 3, 11 / 20 (1961)

7) GOLDSMITH P.L. & WOODWARD R.H. "Cumulative Sum Techniques"
Monograph n° 3
Olivier & Boyd, 1964

8) JOHNSON N.L. "A Simple Theoretical Approach to Cumulative Sum Charts"
Journal of the American Statistical Association, Déc. 1961 - 56 - 296 pages 835 / 840

9) JOHNSON N.L. & LEONE F.C. "Cumulative Sum Control Charts. Mathemetical Principles Applied to their Construction and Use"
Industrial Quality Control - 1962 -
XVIII - 12 - p. 15 / 21
XIX - 1 - p. 29 / 36
XIX - 2 - p. 22 / 28

10) KEMP K.W. "The Average Run Length of the Cumulative Sum Chart when a V-mask is Used"
Journal of the Royal Statistical Society - Serie B - 1961 - 23 - 1, pages 149/ 153

11) LUCAS J.M. "A modified V-Mask Control Schema Technometrics, Vol. 15, Nº 4, nov. 73

12) PAGE E.S. "Continous Inspection Schemes"
Biometrika, vol.41, 1954,p. 100/114

13) PAGE E.S. "Control Charts with Warning Lines"
Biometrika,vol.42,1955,p.243/257

14) PAGE E.S. "On Problems in which a Change in a Parameter Occurs at an Unknown Point"
Biometrika,vol.44,1957,p. 248 / 252

15) PAGE E.S. "Cumulative Sum Charts"
Technometrics 3, 1 - 9, 1961

16) PAGE E.S. "Cumulative Sum Schemes Using Gauging"
Technometrics 4, 97-109, 1962

17) RIDDICK J.H. & GIDDINGS H.W. "Computerized Preparation of Average Cusum Charts for Clinical Chemistry"
Clin. Biochem. 4, 156 / 161, 1971

18) SHEWHART W.A. "In Economic Control of Quality of Manufactured Products"
Van Nostrand, New-York, 1931

19) WETHERILL G.B. "Sampling Inspection and Quality Control"
Methuen 1969.

20) WHITBY L.G. , MITCHELL F.L. & MOSS D.W. "Quality Control in Routine Clinical Chemistry"
In Advances in Clinical Chemistry, Bodansky O. & Stewart C.P. Eds Academic Press, New-York, 1967, vol. 10.

21) Documents relatifs à l'operating System UNIX et au langage C :
- The UNIX Time-Sharing System
- C Reference Manual
- Programming in C - A tutorial
- A tutorial Introduction to the UNIX Text Editor
- UNIX for Beginners.

ANNEXES

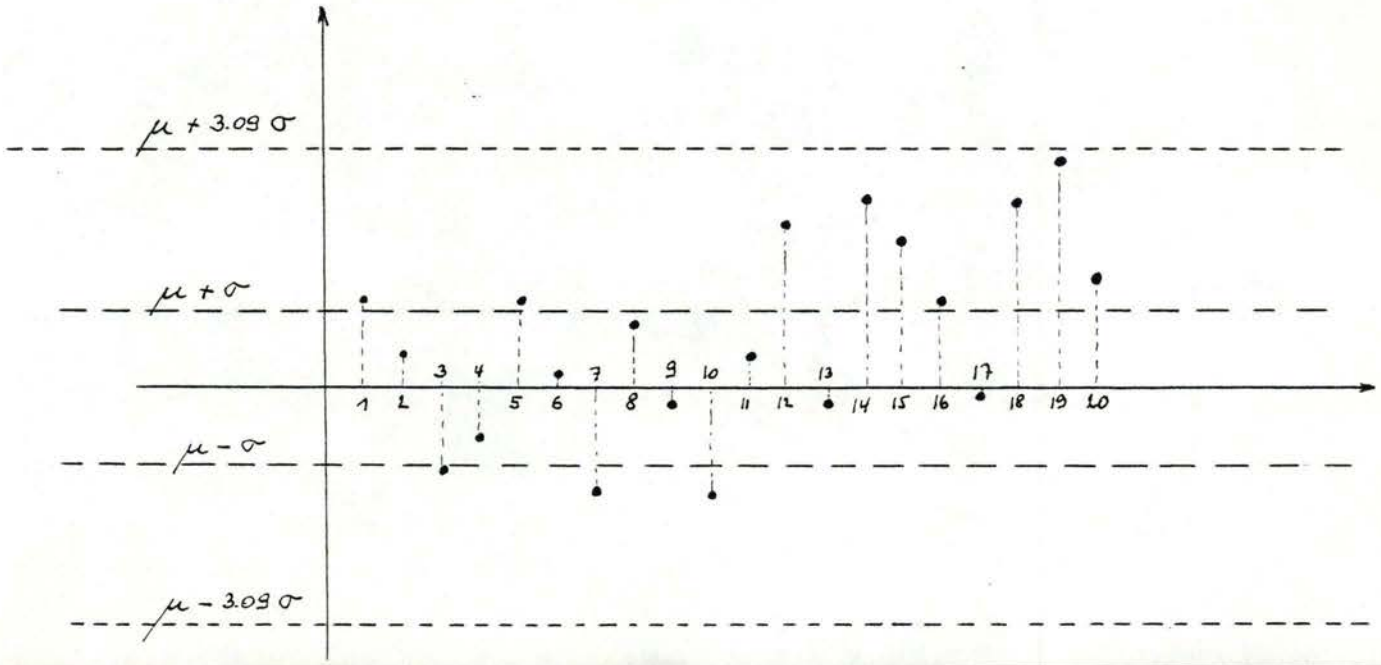
ANNEXE I

-Exemple de la méthode de Shewart
-Exemple de sommes cumulées
-Exemples de V-écrans
-Diagrammes de Goldsmith& Withfield
-Exemple de dispositif basé sur l'intervalle de décision

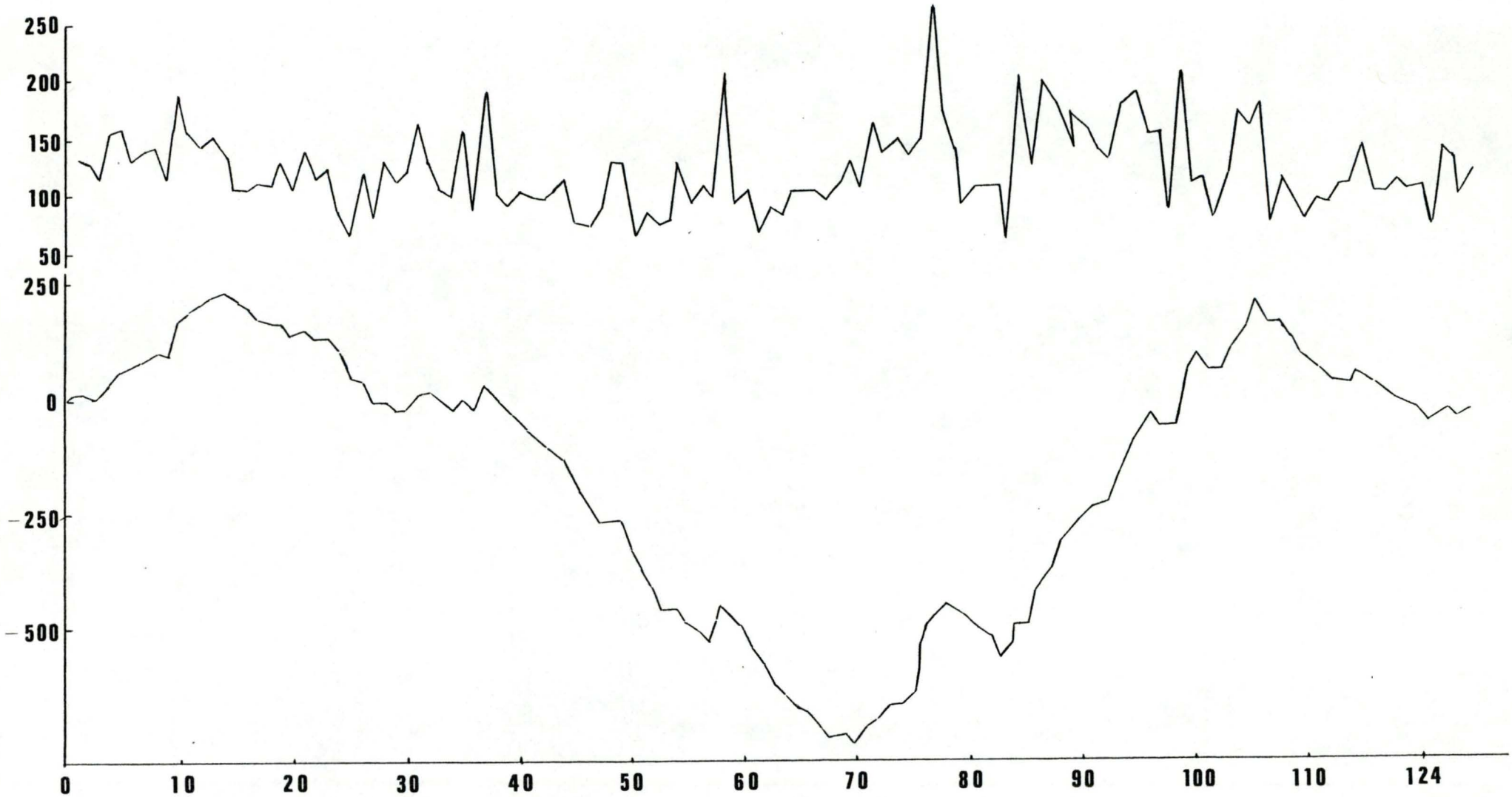
Méthode de Shewart - Exemple numérique.

Soit une population normale de moyenne M = 0 et d'écart-type S = 2 dans laquelle on prélève des échantillons de quatre pièces une par une en remettant la pièce après chaque mesure. On considère les moyennes $\bar{x}_i$ des échantillons. Les dix premiers échantillons suivent une loi normale N (0,I). A partir du onzième, on introduit un dérèglement tel que les échantillons suivent désormais une loi normale N(1.5, 1).

N°	$\bar{x}_i$	N°	$\bar{x}_i$
x1	1.6	x11	0.4
x2	0.4	x12	2.1
x3	-1.1	x13	-0.2
x4	-0.7	x14	2.4
x5	1.6	x15	1.9
x6	0.1	x16	1.1
x7	-1.4	x17	-0.1
x8	0.8	x18	2.4
x9	-0.2	x19	2.9
x10	-1.4	x20	1.4



On voit sur le diagramme que les dix derniers résultats pourtant incorrects sont considérés comme acceptables car figurant à l'intérieur des limites M $\pm$ 3.09S. C'est pour le moins, inacceptable.
Si on ajoute la règle (a) (limites à M $\pm$ S avec 3 points consécutifs en dehors des limites définissant une erreur), on remarque dès lors que les échantillons 14 à 16 consécutifs se trouvent tous les trois au-dessus de la limite, si bien qu'il y a intervention après le seizième échantillon.
Les deux autres règles n'apportent rien de plus.
On n'aura donc plus laissé passer que cinq échantillons incorrects contre un minimum de dix dans le premier cas !
Faut-il se satisfaire de ce résultat ? Le diagramme nous montre qu'il aurait peut-être fallu se rendre compte de la déviation de la moyenne après le quinzième résultat sinon le quatorzième. Cette méthode n'est donc plus à conseiller vu les moyens dont on dispose à l'heure actuelle. En effet, une méthode utilisant les sommes cumulées fera déclencher une action corrective après le quatorzième échantillon, ce qui représente un gain assez appréciable.



SOMMES CUMULEES

Exemple de sommes cumulées.

On possède 124 résultats. La moyenne de ces résultats est égale à 125.68. Le premier schéma représente les résultats bruts obtenus, le second les sommes des écarts à la moyenne

$$\sum_{i=1}^{n} (x_i - \mu).$$

La première figure montre une variabilité très importante des résultats mais il est impossible, en ne considérant que ce schéma de déceler la tendance générale à la hausse ou à la baisse. Si par contre, on construit la deuxième figure avec la somme des écarts à la moyenne, celle-ci étant de 125.68 min.,on remarque instantaténement les variabilités constantes durant certaines périodes.

a) résultats 1 à 14 : diagramme cumulé à une pente croissante, la moyenne des résultats est de 142.

b) résultats 15 à 70 : diagramme cumulé a une pente décroissante, la moyenne n'est plus que de 108.

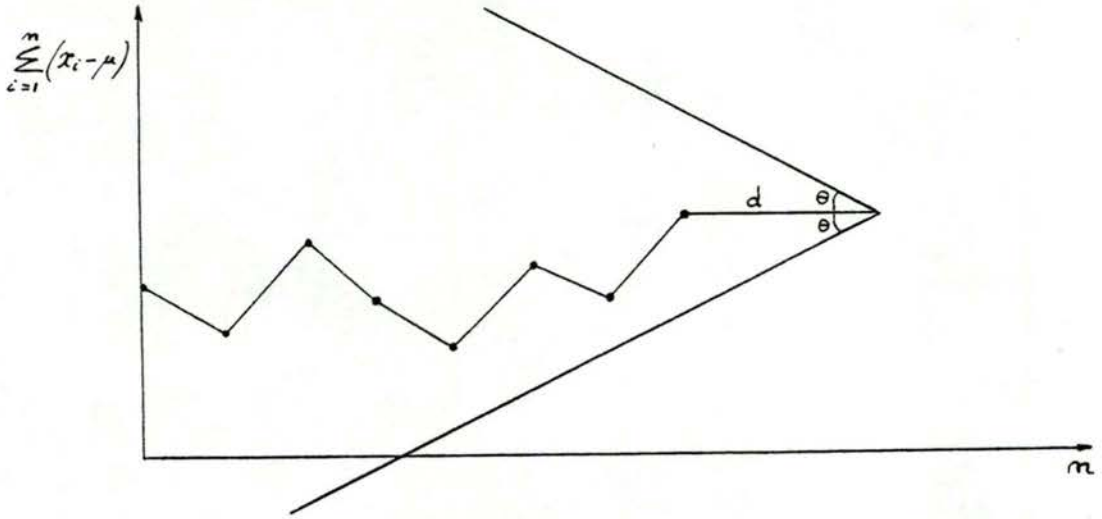
c) résultats 71 à 105 : pente très croissante résultant d'une moyenne très élevée 154.

d) résultats 106 à 124 : pente décroissante, la moyenne est retombée à 113.

Cet exemple montre très bien l'avantage que l'on peut retirer en utilisant un diagramme de sommes cumulées pour effectuer un contrôle de qualité d'uniformité par rapport à la moyenne.

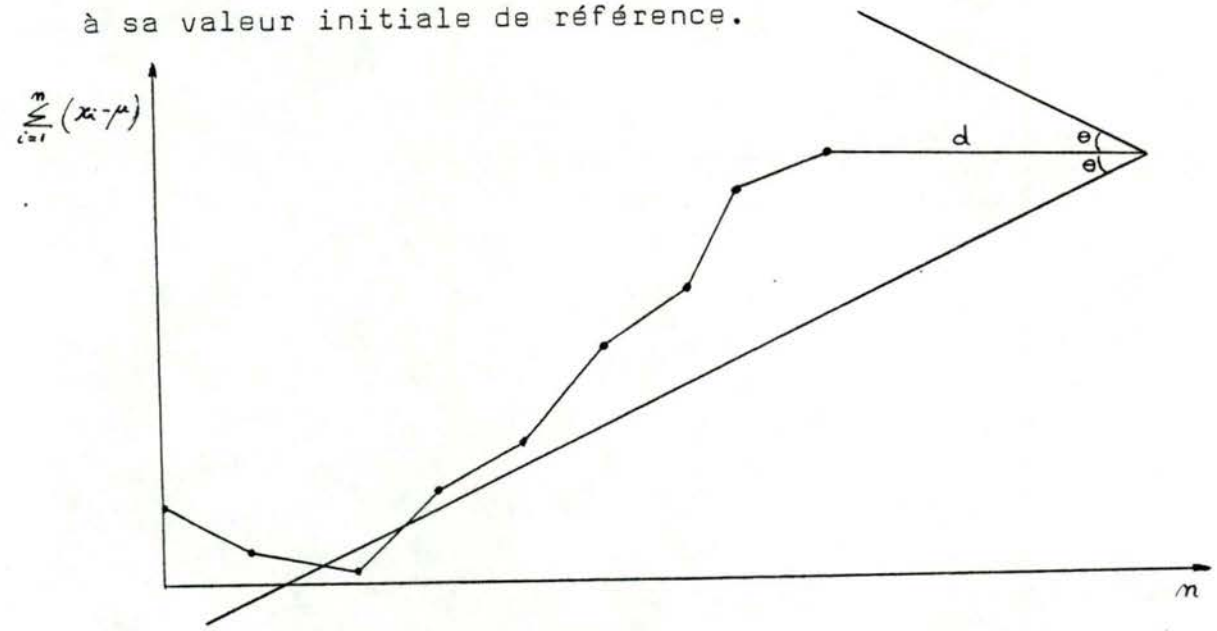
Exemples de V-écrans.

a) Cas où le processus est en état d'équilibre statistique autour de sa moyenne.



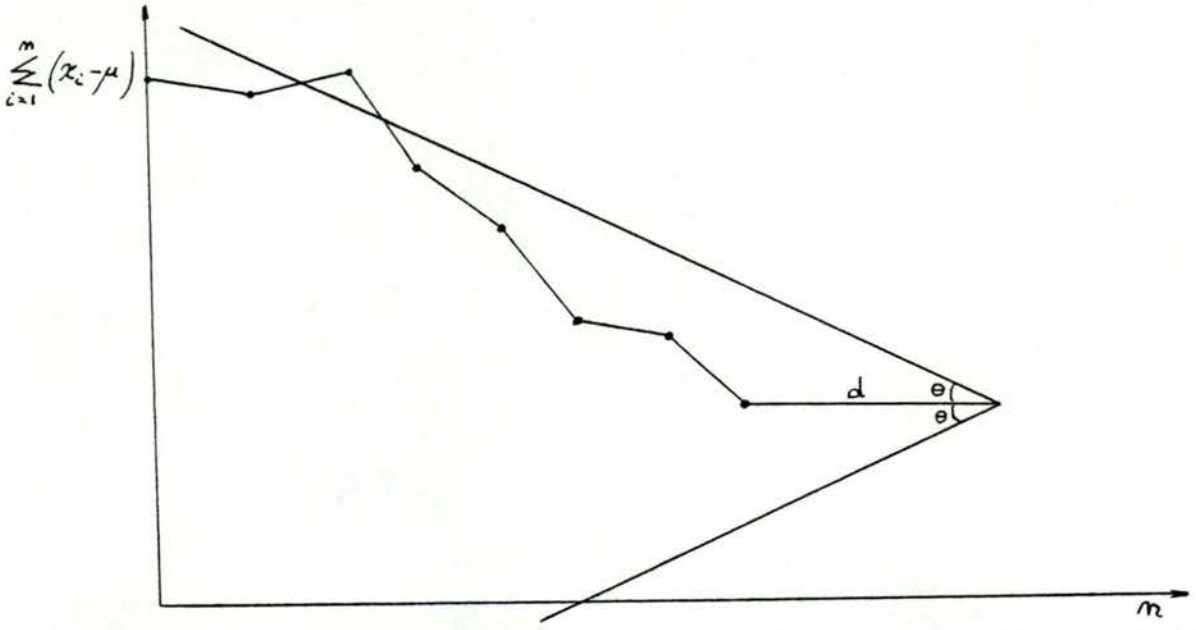
On voit que tous les points sont à l'intérieur des deux branches de l'écran, d'où l'on peut en tirer que le processus suit un développement stationnaire autour de sa moyenne.

b) Cas où le processus voit sa moyenne augmenter par rapport à sa valeur initiale de référence.



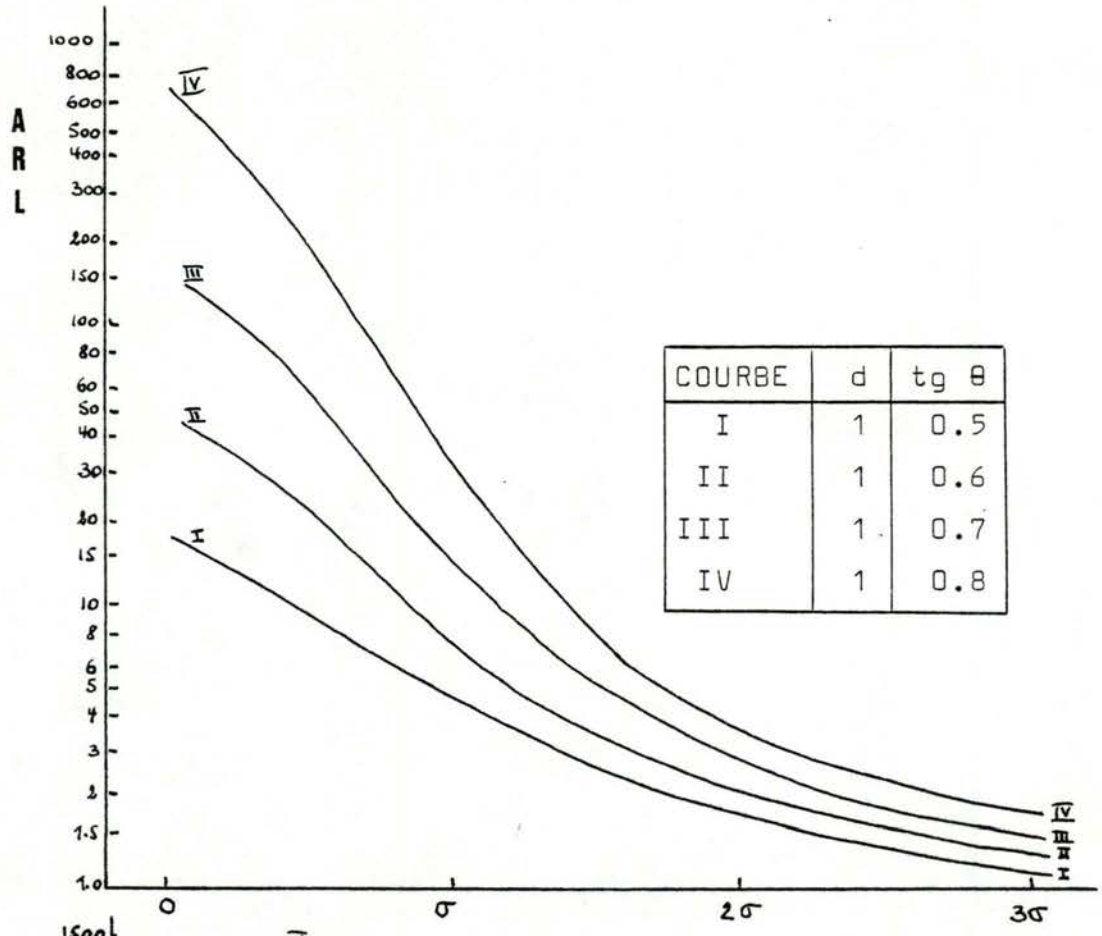
Un point se trouve en dessous de la branche inférieure de l'écran, d'où l'on peut déduire que le processus a augmenté sa moyenne de manière inacceptable. Une action corrective est donc de rigueur.

c) Cas où la moyenne du processus est inférieure à la valeur initiale de départ.

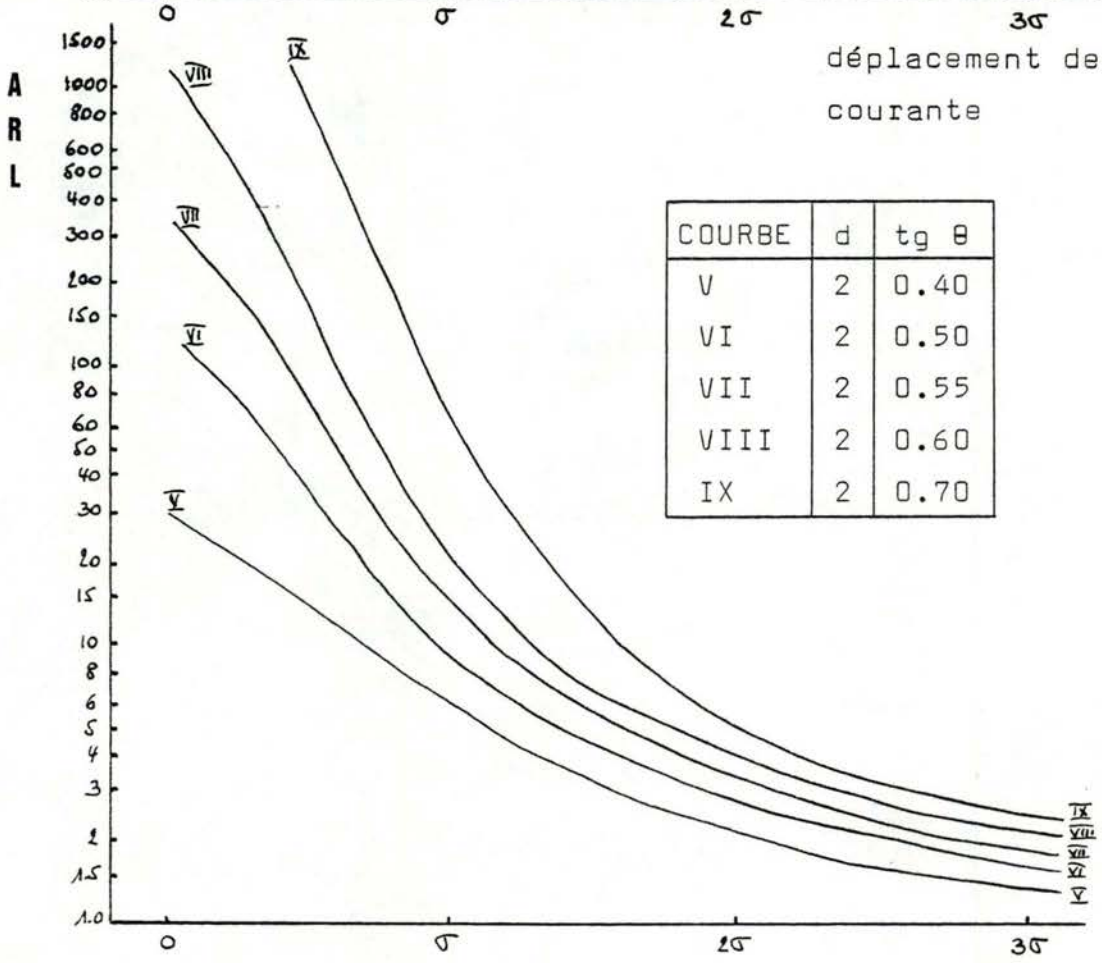


Mêmes conclusions que dans le deuxième cas, mais la sortie se fait ici par le haut, c'est-à-dire décroissance de la moyenne impliquant une action corrective.

Diagrammes de détermination des paramètres d et θ

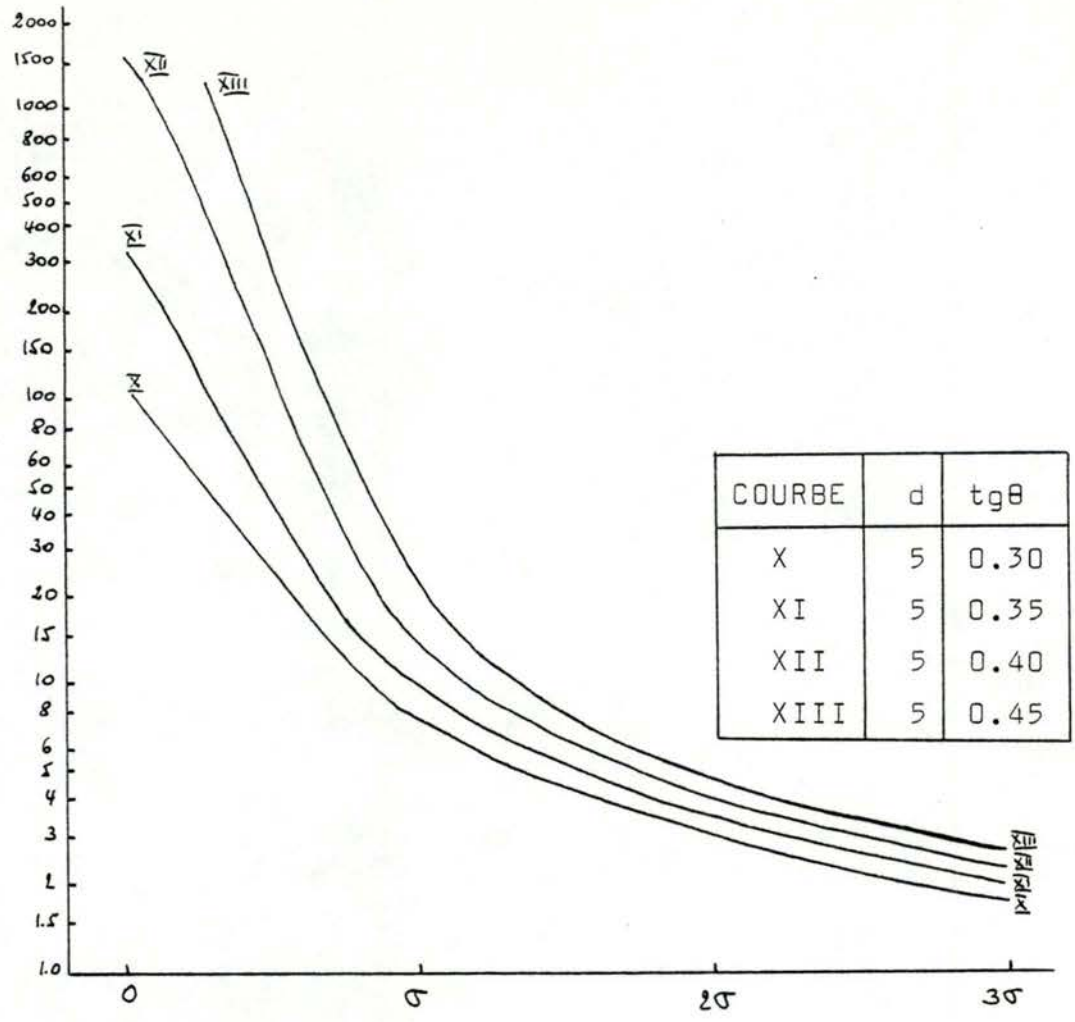


COURBE	d	tg θ
I	1	0.5
II	1	0.6
III	1	0.7
IV	1	0.8



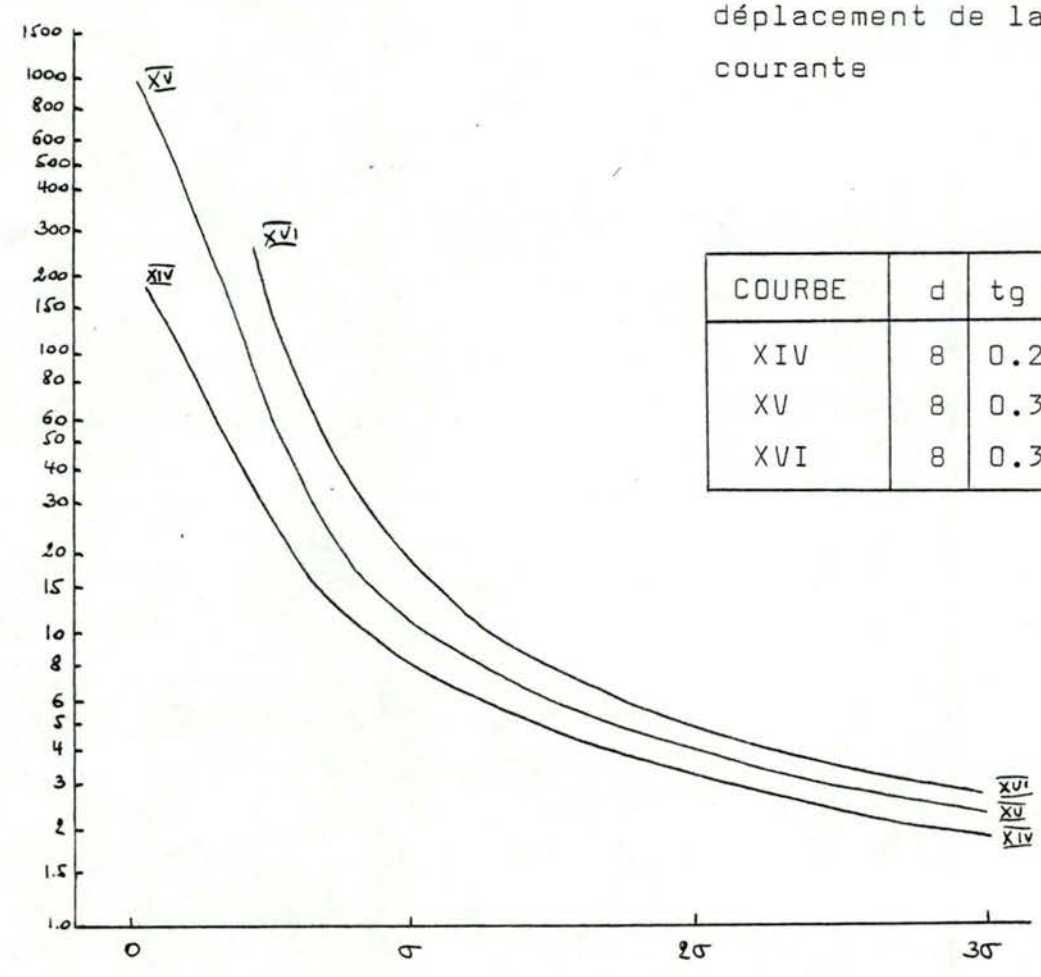
déplacement de la moyenne courante

COURBE	d	tg θ
V	2	0.40
VI	2	0.50
VII	2	0.55
VIII	2	0.60
IX	2	0.70



COURBE	d	tgθ
X	5	0.30
XI	5	0.35
XII	5	0.40
XIII	5	0.45

déplacement de la moyenne courante



COURBE	d	tg θ
XIV	8	0.25
XV	8	0.30
XVI	8	0.35

Dispositif basé sur intervalle de décision.

Diagramme standard

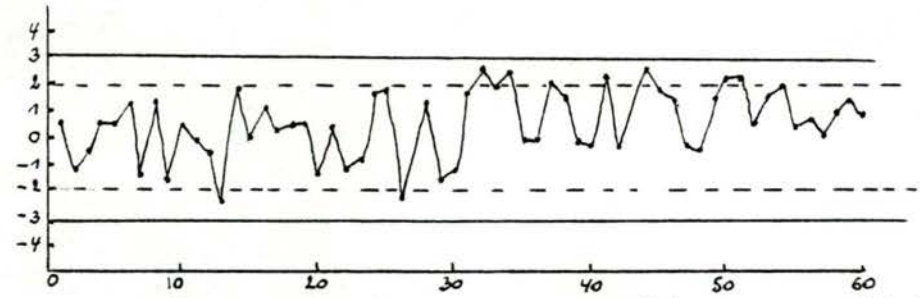


Diagramme de sommes cumulées rapporté à la valeur centrale 0.5

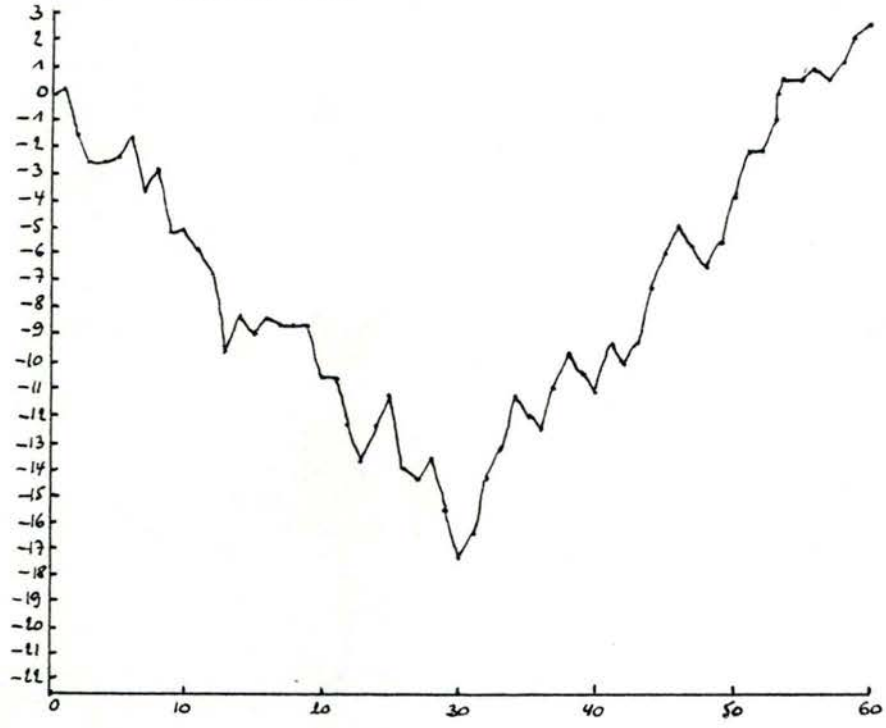
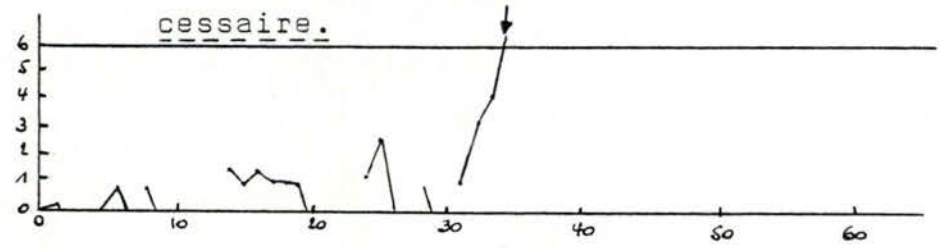


Diagramme de sommes cumulées représenté en tant que nécessaire.



Les 30 premiers résultats suivent une loi normale (0,1)

Les 30 résultats suivants suivent une loi normale (1,1)

La valeur centrale de référence k est choisie égale à 0.5

Sur le diagramme de sommes cumulées représenté en tant que nécessaire, on essaie de détecter le plus rapidement possible les accroissements dans la moyenne du processus. On choisit pour cela la valeur centrale de référence k à mi-chemin entre M_0 (=0) et le niveau qui apparaît comme juste insatisfaisant. On ne reporte alors sur le diagramme de sommes cumulées que les valeurs qui dépassent k. Si la somme cumulée revient à 0, on en déduira que le processus tend de nouveau vers sa moyenne (ou plus bas puisque cela ne nous intéresse pas) mais si elle atteint ou dépasse l'intervalle de décision h (ici = 6), on pourra tirer la conclusion que la moyenne courante du processus a augmenté de façon inacceptable.

ANNEXE II

-construction d'une population
théorique normale
Méthode de Box et Muller
-Données à traiter
-Application des 3 méthodes sur
ces données, paramètres respectifs

Construction d'une population théorique normale.

Méthode de Box et Muller

On trouvera la démonstration de cette méthode dans le cours de Simulation de Mr Fichefet, chap.III,2,5.

Si l'on veut déterminer deux valeurs artificielles x_1 et x_2 d'une variable $X = N(m, \sigma)$, il suffit de déterminer deux nombres aléatoires u_1 et u_2 suivant une distribution uniforme sur $[0,1]$ et d'écrire :

$$x_1 = m + (-2\sigma^2 \ln u_1)^{\frac{1}{2}} \cos 2\pi u_2$$

$$x_2 = m + (-2\sigma^2 \ln u_1)^{\frac{1}{2}} \sin 2\pi u_2$$

Dans notre cas, on se propose de construire une population suivant une loi normale N (0,1)

On a donc : $\sigma = 1$

$m = 0$

d'où : $x_1 = (-2 \ln u_1)^{\frac{1}{2}} \cos 2\pi u_2$

$x_2 = (-2 \ln u_2)^{\frac{1}{2}} \sin 2\pi u_2$

On peut se limiter à un échantillon de taille 50 ; on pourra déjà se faire une très bonne estimation des longueurs moyennes de séries que l'on obtiendra ! Pour calculer les ARL (2σ), il suffira de forcer un écart de 2 par rapport à la moyenne (ajouter 2 à tous les résultats) pour simuler une augmentation de la moyenne courante du processus de deux écarts-types .

Population obtenue par la méthode de Box et Muller.

-0.71	0.14	-0.40	0.49	-0.92	-0.59	0.33	-0.28
-0.34	1.76	-0.65	-0.79	0.22	-1.14	0.13	0.39
-0.90	0.49	-0.23	-1.19	-0.53	0.49	-0.10	0.52
2.52	-0.82	-1.64	-0.31	-1.89	-0.48	-0.15	1.22
0.86	-0.81	0.12	-0.64	0.84	-1.01	-0.10	1.55
0.06	-0.24	-0.25	0.53	-0.33	0.70	-0.52	-0.94
-0.98	-2.09						

Données modifiées.

1.29	2.14	1.60	2.49	1.08	1.41	2.33	1.72
1.66	3.76	1.35	1.21	2.22	0.86	2.13	2.39
1.10	2.49	1.77	0.81	1.47	2.49	1.90	2.52
4.52	1.18	0.36	1.69	0.11	1.52	1.85	3.22
2.86	1.19	2.12	1.36	2.84	0.99	1.90	3.55
2.06	1.76	1.75	2.53	1.67	2.70	1.48	1.06
1.02	-0.09						

On peut donc maintenant effectuer un contrôle de qualité sur ces données. On considérera toujours la moyenne du processus égale à zéro et son écart-type égal à un. On va voir combien de coups sont nécessaires en moyenne pour chaque méthode pour détecter l'augmentation de la moyenne de deux écarts-types. Cela nous fournira une bonne estimation de ARL (2σ) que l'on pourra comparer avec celle théorique qui permet de se fixer les paramètres de décision (d, θ ou h)

Application de la méthode 1

$\mu = 0 \quad \sigma = 1 \quad w = 1 \quad \Delta\mu = 2\sigma = 2$ ARL $(\Delta\mu) = 3$

$tg\,\theta = \frac{1}{2}\,\frac{\Delta\mu}{w} = 1$ ARL $(0) = 250$

$\frac{d}{w} = 2.20 \longrightarrow d = 2.20$

on a donc : $d\ tg\,\theta = 2.20$

$w\ tg\,\theta = 1$

$\mu = 0$

Sorties après : 4,4,2,5,3,4,2,1,7,3,4,1,3,3 coups

Application de la méthode 2

$\mu = 0 \quad \sigma = 1 \quad w = 1 \quad \Delta\mu = 2\sigma = 2$ ARL$(\Delta\mu) = 3$

ARL$(0) = 150$

choix de la courbe XIV

$d = 8 \times 1 = 8$

$tg\,\theta = 0.25 \times 2 = 0.50$

on a donc : $d\ tg\,\theta = 4$

$w\ tg\,\theta = 0.5$

$\mu = 0$

Sorties après : 4,4,2,5,3,4,3,7,3,4,2,3,3 coups

Application de la méthode 3

$\mu = 0 \quad \sigma = 1 \quad w = 0 \quad \Delta\mu = 2\sigma = 2$ ARL$(\Delta\mu) = 3$

ARL $(0) = 380$

$$h_{lu} = 2.2$$

$$h = 2.2 \times 2 \times 1 = 4.4$$

On a donc: $d \ tg\theta = h = 4.4$

$w \ tg\theta = 0$

$\mu = 0$

Sorties après : 3,3,3,2,4,3,4,2,1,5,2,3,3,2,3,3 .

ANNEXE III

-Contrôle de qualité sur HP 25
 Organigramme
 Programme
-Populations de résultats et paramètres
 .lipides totaux normaux
 .lipides totaux pathologiques
 .bilirubine totale normale
 .bilirubine totale pathologique

Programme de contrôle de qualité

Les essais préliminaires pour essayer d'évaluer l'utilité des différentes méthodes de contrôle de qualité ont été effectuées sur une simple machine à calculer programmable par clavier (Hewlett-Packard 25).

Il m'a semblé intéressant de publier la manière dont le contrôle a été programmé car ce sera la même méthode qui sera utilisée dans la future implémentation en machine (en l'occurrence PDP-II/70).

Le problème dans ce cas-ci est la faible taille de la mémoire (seulement 8 registres) impliquant l'introduction des données une à une. Une étude sur un grand nombre de résultats représente donc un travail long et fastidieux. C'est pourquoi il a semblé indispensable de traiter le problème par ordinateur où l'on pourra évidemment stocker toutes les données qui nous seront utiles et nécessaires sans les recalculer à chaque fois.

Programme de HP 25

8 registres : R0 : accumulateur 2
R1 : moyenne des résultats = μ
R2 : $x_i - \mu - w \operatorname{tg}\theta$
R3 : $x_i - \mu + w \operatorname{tg}\theta$
R4 : accumulateur 1
R5 : écart $w \operatorname{tg}\theta$
R6 : h ou $d \operatorname{tg}\theta$ (intervalle de décision)
R7 : nombres de données générées avant détection d'erreur.

x_i = donnée à analyser.

On a une pile opérationnelle à 4 positions X,Y,Z,T superposées dont la position X est toujours affichée. La machine utilise la notation polonaise inversée.

Initialisation

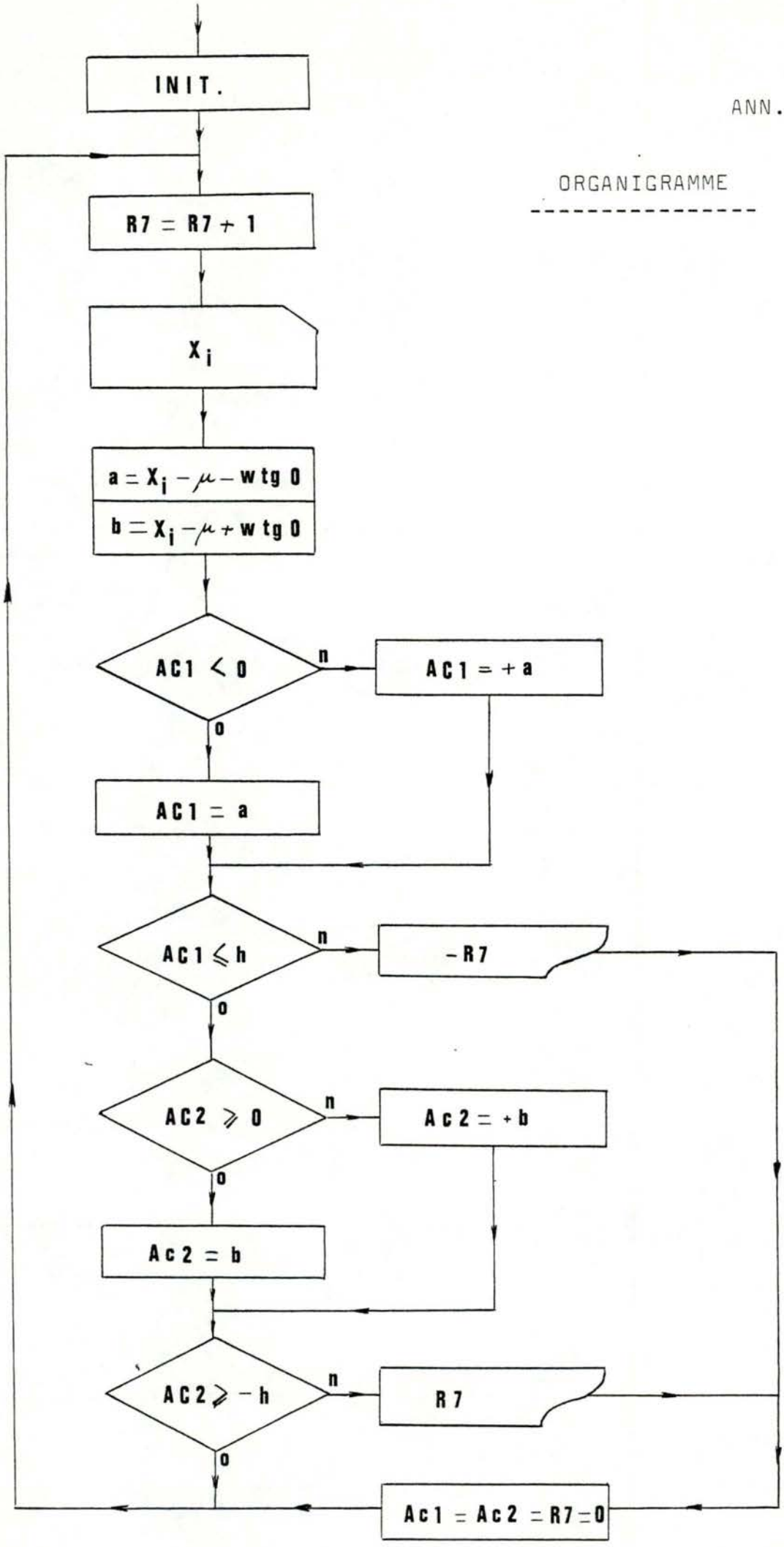
On calcule la moyenne et on la met dans R1. On calcule l'écart-type, on choisit la méthode de détermination des paramètres d et θ, on calcule w tgθ qu'on met dans R5 et d tgθ qu'on met dans R6.

Exécution

On introduit la donnée ; on exécute le programme. Si l'écran affiche 0.00, la donnée est considérée comme correcte, on peut rentrer la suivante et ainsi de suite.
Si l'écran affiche un certain nombre différent de 0, il s'agit du nombre de données générées avant détection d'erreur (R7). Cette dernière donnée est donc refusée et considérée comme incorrecte. Si le nombre affiché est positif, il dénote d'une diminution dans la moyenne courante du processus et inversément, un affichage négatif indique un accroissement du processus. En cas de détection d'erreur, les accumulateurs sont remis à zéro ainsi que le compteur de données !

On trouvera page ANN.III.3 l'organigramme et page ANN.III.4 le programme lui-même.

ORGANIGRAMME



Programme HP 25

```
 1.   1
 2.   STO + 7
 3.   CLX
 4.   R/S
 5.   RCL 1
 6.    -
 7.   RCL 5
 8.    -
 9.   STO 2
10.   RCL 5
11.   2
12.   X
13.   +
14.   STO 3
15.   f STK
16.   RCL 2
17.   RCL 4
18.   g  x < 0
19.   R↓
20.   +
21.   STO 4
22.   RCL 6
23.   f  X ≥ Y
24.   GTO 28
25.   RCL 7
26.   CHS
27.   GTO 40
28.   f STK
29.   RCL 3
30.   RCL 0
```

```
31.   g X ≥ 0
32.   R↓
33.   +
34.   STO 0
35.   RCL 6
36.   CHS
37.   f  X < Y
38.   GTO 01
39.   RCL 7
40.   R/S
41.   0
42.   STO 0
43.   STO 4
44.   STO 7
45.   GTO 01
```

Populations de résultats réels. Laboratoire du docteur Noel, juillet-août 1978

Lipides totaux normaux

560	570	560	580	580	580	580	580	580	580
590	590	580	570	570	590	600	600	580	600
570	580	590	580	570	600	600	580	590	590
580	590	590	580	580	590	600	590	580	590
590	570	580	590	580	600	590	590	590	
590	580	580	580	590	580	580	600	580	
610	590	590	580	580	600	590	580	590	
590	570	580	580	590	570	600	600	580	
580	580	570	590	580	590	580	590	590	
600	570	580	580	580	590	600	590	600	
590	570	590	590	580	600	590	590	580	
580	590	570	570	570	590	600	600	590	
590	570		590	590	600	580	590	600	

120 observations

μ = 585.33 ARL (2σ) = 3

σ = 9.87

Méthode 1

w = 10

d tg θ = 21.91

w tg θ = 9.87

μ = 585.33

aucune sortie !

Méthode 2

w = 10 courbe XIV d = 8 x 9.87 = 78.96

tg θ = 0.25 x 2 x 9.87/10 = 0.49

d tg θ = 38.97

w tg θ = 4.9

μ = 585.33

sortie après 27 données (560) !

Méthode 3

h_{lu} = 2.2 h = 2.2 x 2 x 9.87 = 43.43

w tg θ = 0

μ = 585.33

sorties après -10,12,5,12,11,6,-15,-8,-6,-11,-7,-16 données.

Méthode 3 avec ARL (2σ) = 4

h_{lu} = 3.1 h = 3.1 x 2 x 9.87 = 61.19

w tg θ = 0

μ = 585.33

sorties après 23,12,15,13,-13,-9,-13,-20 données.

Lipides totaux pathologiques

620	600	600	580	590	600	610	580	600	590
610	610	590	590	580	590	590	620	600	600
620	600	580	570	590	600	600	580	590	610
600	590	580	590	570	600	600	600	600	
620	590	570	570	580	580	590	610	600	
580	600	590	580	570	600	600	590	610	
600	590	590	590	580	600	600	600	590	
600	600	600	600	580	590	590	610	600	
600	590	590	590	590	600	600	590	590	
610	580	590	590	590	600	600	600	600	
600	580	580	580	600	600	580	590	590	
590	600	580	590	600	580	580	590	580	
610	590	570	580	590	600	610	600	600	

120 observations

μ = 593.33 ARL (2σ) = 3

σ = 11.2521

Méthode 1

$d \operatorname{tg} \theta = 24.98$

$w \operatorname{tg} \theta = 11.2521$

$\mu = 593.33$

sorties après -3,41,14 données.

Méthode 2

courbe XIV

$d \operatorname{tg} \theta = 50.63$

$w \operatorname{tg} \theta = 5.6$

$\mu = 593.33$

sorties après - 3,37,16 données

Méthode 3

$h = 49.50$

$w \operatorname{tg} \theta = 0$

$\mu = 593.33$

sorties après -3,-7,-6,14,8,4,8,6,3,-20,-19,-12 données

Méthode 3 avec ARL (2σ) = 4

$h = 69.7630$

sorties après -3,-10,18,9,6,-23,-20 données

Bilirubine totale normale

.41	.56	.55	.48	.32	.43	.41	.38	.51	.46
.57	.78	.48	.50	.46	.42	.38	.44	.42	.34
.46	.49	.51	.36	.37	.39	.38	.41	.37	.42
.68	.62	.46	.41	.43	.27	.36	.40	.34	.49
.81	.57	.40	.45	.39	.35	.32	.41	.39	
.57	.52	.41	.40	.24	.30	.24	.43	.40	
.39	.37	.44	.48	.33	.37	.45	.44	.46	
.41	.56	.58	.45	.34	.49	.37	.41	.29	
.47	.63	.61	.48	.48	.45	.40	.49	.32	
.59	.50	.51	.42	.57	.33	.47	.44	.41	
.42	.39	.47	.46	.45	.42	.43	.47	.48	
.42	.49	.49	.43	.38	.38	.49	.41	.38	
.70	.57	.57	.44	.27	.44	.40	.47	.42	

121 observations

μ = .44479 ARL (2σ) = 3

σ = .09

Méthode 1

w = 1

d tgθ = 0.20

w tgθ = 0.09

μ = 0.445

sorties après -5,-10 données

Méthode 2

courbe XIV

d tgθ = 0.0366

w tgθ = 0.0478

μ = 0.445

sorties après -2,-2,-1,-1,-4,-3,.....données

inacceptable : beaucoup trop sensible!

Méthode 3

h = 0.4212

sorties après -5,-9,-3,-5,-12,-5,19,11,8,7,24

8 données

Méthode 3 avec ARL (2σ) = 4

sorties après -5,-1I,-7,-13,24,11,12,13,19 données.

Bilirubine totale pathologique

6.90	7.32	6.67	7.27	5.35	4.58	6.81	5.99
8.32	6.21	6.66	7.32	5.38	4.49	5.88	5.79
6.5	5.85	8.75	7.68	6.59	5.53	6.21	5.94
5.7	5.68	6.72	5.18	6.04	5.51	7.34	5.68
10.04	7.67	6.87	6.34	5.19	8.2	6.77	5.89
6.36	7.08	6.69	6.27	5.31	7.6	8.27	6.19
5.73	6.8	7.04	6.49	6.04	8.03	6.81	5.86
5.95	6.16	6.81	6.47	5.87	7.77	6.07	6.92
6.17	5.69	6.64	6.69	7.34	7.82	7.06	6.09
7.2	6.51	6.50	4.24	7.51	7.60	6.38	
7.67	6.48	6.24	3.83	6.77	7.22	5.90	
8.81	5.96	7.05	5.37	7.75	7.42	5.74	
8.27	6.06	8.03	8.46	3.86	6.98	6.31	
7.66	6.39	6.94	8.14	4.57	6.42	5.66	
7.61	7.12	7.55	6.56	6.89	6.83	6.09	
6.39	7.03	6.41	5.24	5.71	7.16	5.71	

121 observations

μ = 6.5715 ARL (2σ) = 3

σ = 1.0264

Méthode 1

w=1

d tg θ = 2.28

w tg θ = 1.0264

μ = 6.5715

Sorties après -5,54,19 données

Méthode 2

courbe XIV

d tg θ = 4.2140

w tg θ = 0.5132

μ = 6.5715

Sorties après -14,47, 27,21,-9 données

Méthode 3

h = 4.52

sorties après -5,-8,-27,-10,9,10,9,4,6, -8,
17 données

Méthode 3 avec ARL(2σ) = 4

h = 6.3643

sorties après -12,-32,14,11,11,-8,27 données

ANNEXE IV

-INIT : organigramme et programme.

-AJOUT : organigrammes et programme.

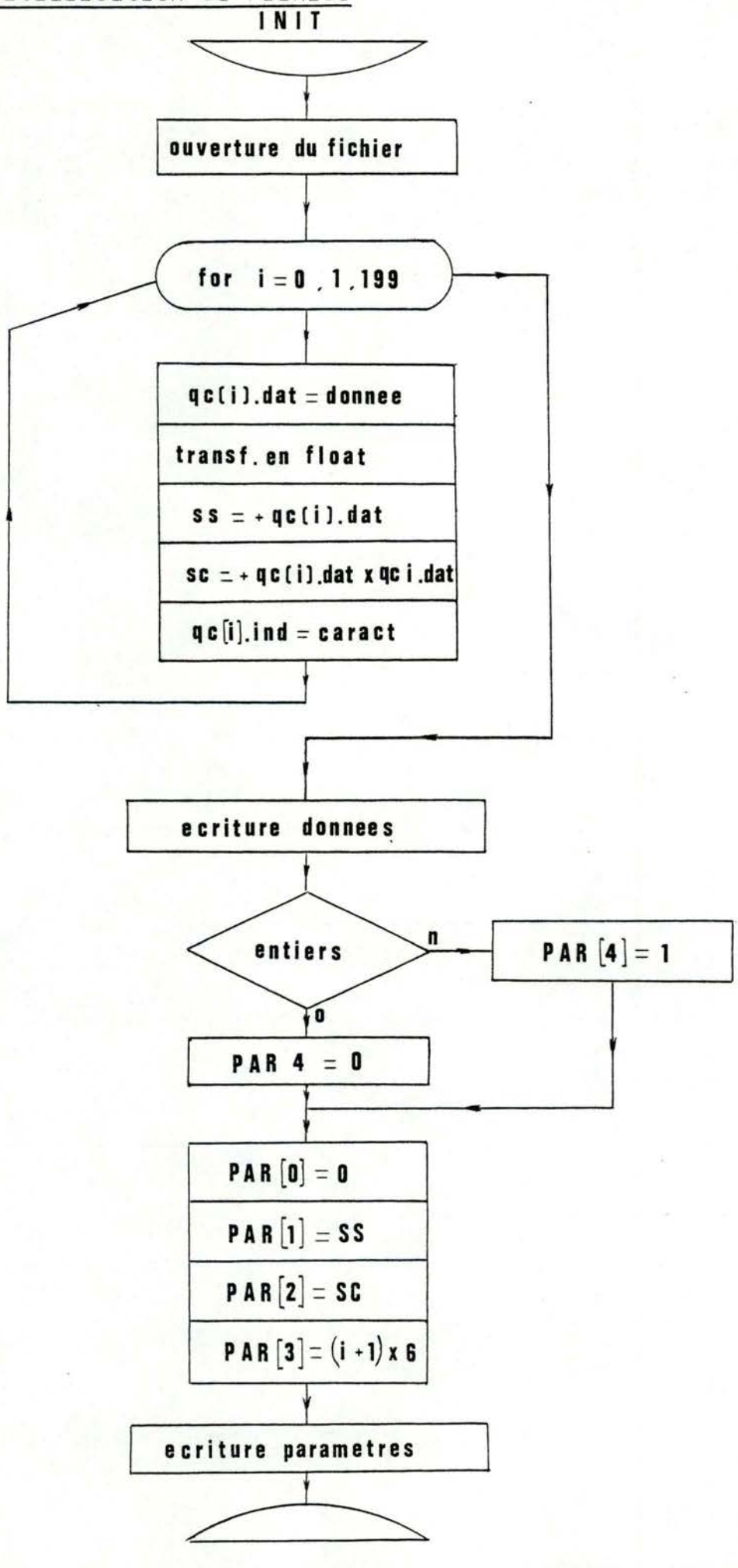
-QC : organigrammes et programme.

-Exemple sur données du laboratoire.

Création du fichier : CREAT

```
                    /*  creat est un programme qui
                        cree un fichier au nom que l'on
                        donne ( 10 caracteres maximum ).
                    */

main() {
            char *fil;
            int dataf,n;
        printf("  filename =  ");
        n = read(0,fil,10);
        fil[n-1] = '\0';
        dataf = creat(fil,0777);
        close(dataf);
           }
```

Initialisation du fichier



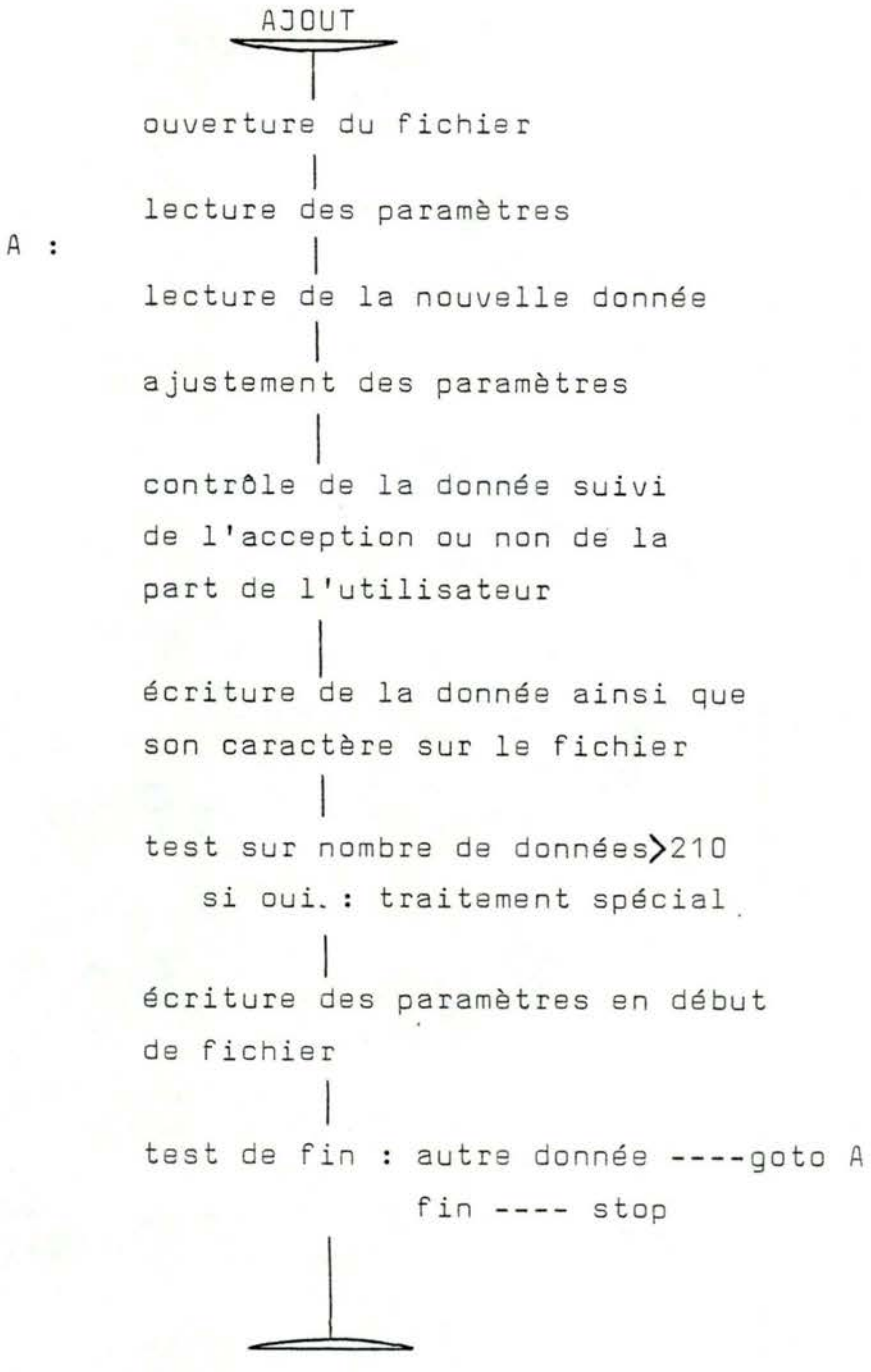
```
cat init.c
                        /*  init est un programme qui
                            initialise un fichier .
                            il s'agit d'y rentrer un nombre
                            de donnees inferieur a 200 avec
                            leur caractere ( 1 si donnee
                            correcte ,0 sinon ).
                            l'utilisateur devra encore dire
                            si les donnees apparaissant dans
                            ce fichier sont entieres ou non.
                        */

double atof();
main () {
      char *data;
      char *fil;
      float par[10];
      float ss,sc;
      char *rep;
      char don[4];
    int i,J,n,dataf;
    struct   {
          float dat;
          char car; } ac[200];

                        /*  ouverture du fichier  */

      printf(" filename : ");
      n = read(0,fil,10);
      fil[n-1] = '\0';
      dataf = open(fil,2);
      if(dataf == -1) {   printf(" wrong filename !!");
                          exit();
                      }

                      /* lecture des donnees et des
                         caracteres
                      */

      for (i = 0;i < 200;i++ )  {
         printf(" is there a data ? y or n?");
         read(0,rep,4);

         if(rep[0] != 'y')  break;

           J = i + 1;
           printf(" donnee numero %d= ",J);
           n = read(0,data,8);
           ac[i].dat = atof(data);
           ss =+ ac[i].dat;
           sc =+ ac[i].dat * ac[i].dat;
           printf("      caractere numero %d=",J);
           n = read(0,don,4);
           ac[i].car = don[0];
                          }
```

```
                    /*  ecriture des donnees
                        et de leurs caracteres a partir
                        du byte 40 .
                    */

seek(dataf,40,0);
n = write(dataf,ac,J * 6);


                     /* calcul des parametres */

   par[0] = 0;
   par[1] = ss;
   par[2] = sc;
   par[3] = J * 6;

 resp:
 printf("inteser ? y or n ?");
 read(0,rep,4);
 switch(rep[0]) {

       case 'y':
                 par[4] = 0;
                 break;

       case 'n':
                 par[4] = 1;
                 break;

       default:
                 goto resp;
                 break;

                  }

              /*   ecriture des parametres  */

   seek(dataf,0,0);
   write(dataf,par,40);

 close(dataf);
   }
```

Programme AJOUT

Organigrammes.

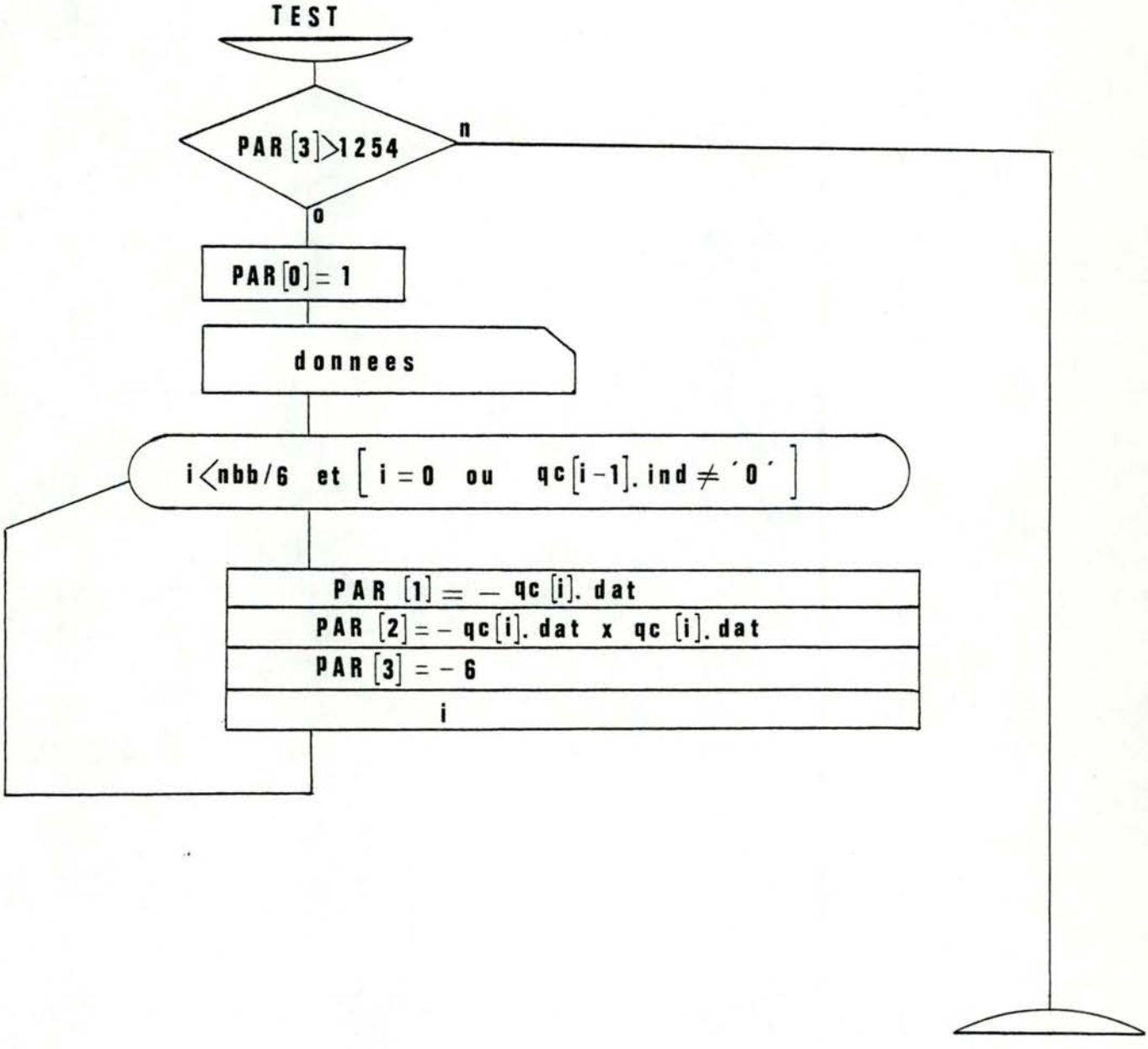
Il s'agit d'ajouter une donnée au fichier, d'en effectuer le contrôle de qualité et ensuite de préparer le travail de qc en ne gardant au maximum que les 210 dernières données du fichier.



Acceptation

Si la donnée est reconnue correcte (incorrecte) par le programme, l'utilisateur a le droit de refuser le verdict et de forcer par la même occasion le caractère qui lui semble le meilleur. En cas de refus d'une solution, il est conseillé d'exécuter QC pour ajuster les paramètres de telle sorte que le résultat sur la dernière donnée satisfasse mieux l'utilisateur.

Test Si on a plus de 210 données retenues, voici le traitement effectué pour réduire ce nombre.
Ce test assurera que l'on commencera toujours un contrôle de qualité après une donnée incorrecte, c'est-à-dire que les accumulateurs seront nuls.



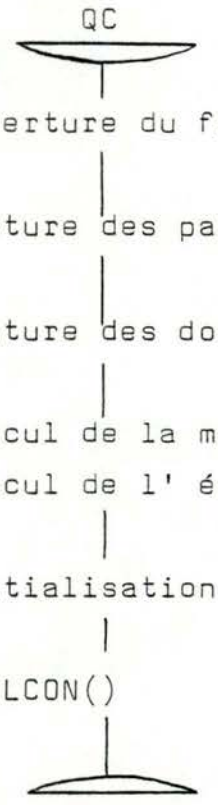
```
cat ajout.c
                        /* ajout : ajouter une donnee a un fichier
                           apres en avoir fait le controle de
                           qualite , preparer le travail d'une
                           execution future de ac en ne conser-
                           vant qu'au maximum les 210 dernieres
                           donnees .
                        */

double atof();
main() {
     struct {  float dat;
               char ind; } ac[210];
     float par[10];
     char *don,*rep;
     char *fil;
     float a,b;
     int dataf,n,i,nbb;

               /* ouverture du fichier */

     printf("  filename : ");
     n = read(0,fil,10);
     fil[n-1] = '\0';
     dataf = open(fil,2);
     if( dataf == -1 ) {   printf("wrong filename !!");
                           exit();
                       }
     read(dataf,par,40);

               /* lecture de la donnee */
lect:
     printf("  the data is : ");
     read(0,don,10);
     ac[0].dat = atof(don);

               /* ajustement des parametres */

     par[1] =+ ac[0].dat;
     par[2] =+ ac[0].dat * ac[0].dat;
     par[3] =+ 6;
     nbb = par[3];
```

```
                /* controle de qualite */

        a = ac[0].dat - par[9] - par[5];
        b = ac[0].dat - par[9] + par[5];

        if ( par[6] >= 0 )  par[6] =+a;
                           else par[6] = a;

        if ( par[6] > par[8] )    {   par[6] = par[7] = 0;
                                      ac[0].ind = '0';
                                  }

           else    {      if ( par[7] >= 0 )   par[7] = b;
                                          else par[7] =+ b;

                          if ( par[7] < -par[8] )    {  par[6] = par[7] = 0;
                                                        ac[0].ind = '0';
                                                     }
                                  else    ac[0].ind = '1';
                   }


        printf(" 1 = bon , 0 = incorrect  :  %c\n",ac[0].ind);


                /* acceptation de la solution */

resp:
        printf(" do you accept the solution ? y or n ?");
        read(0,rep,2);

        if ( rep[0] == 'n' )  { if(ac[0].ind == '1' )  ac[0].ind = '0';
                                       else  ac[0].ind = '1';
                                printf("now,you must execute ac.o\n");
                              }

         else  if (rep[0] != 'y')  {  printf(" wrong respons!\n");
                                      goto resp;
                                   }

                /* ecriture de la donnee */

        seek(dataf,0,2);
        write(dataf,ac,6);
```

```
                    /* test sur nombre de donnees */

      if ( par[3] > 1254 ) {    par[0] = 1;
                                seek(dataf,-nbb,1);
                                n = read(dataf,ac,nbb);
printf("premiere donnee = %f\n",ac[0].dat);
            while((i < (nbb / 6)) && ((i == 0) || ac[i-1].ind != '0'))

                                    {  par[3] -= 6;
                                       par[1] += ac[i].dat;
                                       par[2] += ac[i].dat * ac[i].dat;
                                       i++;
                                    }
                              }


                             /* test de fin  */

      printf(" is there another data  ? y or n ?");
      read(0,rep,2);

      if(rep[0] == 'y') goto lect;

                          /* ecriture des parametres */

      seek(dataf,0,0);
      write(dataf,par,40);

      close(dataf);

}
%
```

Programme QC

Organigrammes.



Le sous-programme QUALCON() de contrôle de qualité est basé sur le même organigramme que celui déjà proposé pour la HP25. L'initialisation équivaut à donner au h_{lu} la plus petite valeur considérée ainsi qu'à son incrément incr.

intervalle de décision$_{lu}$ = 1.9

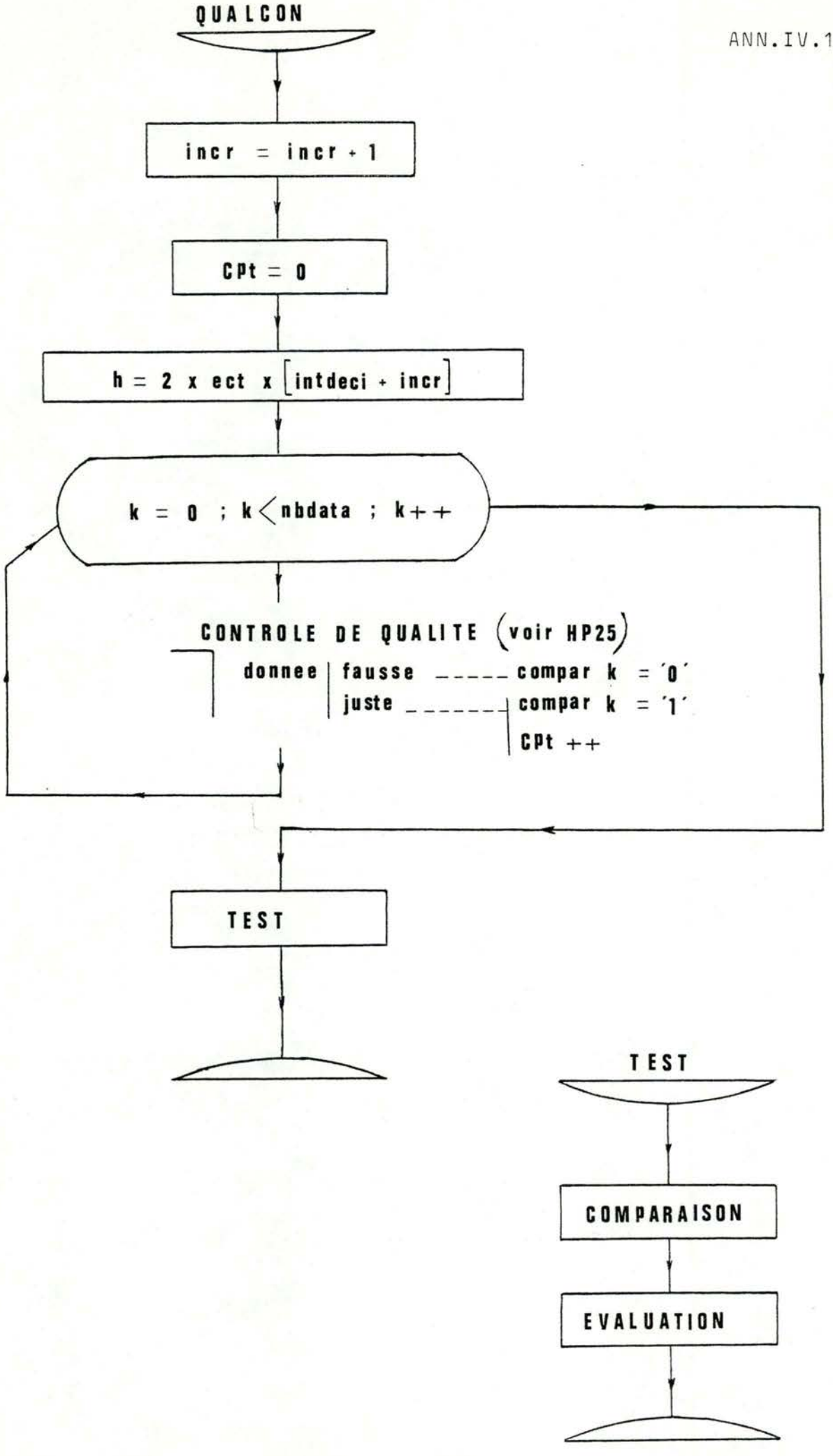
incr = 0

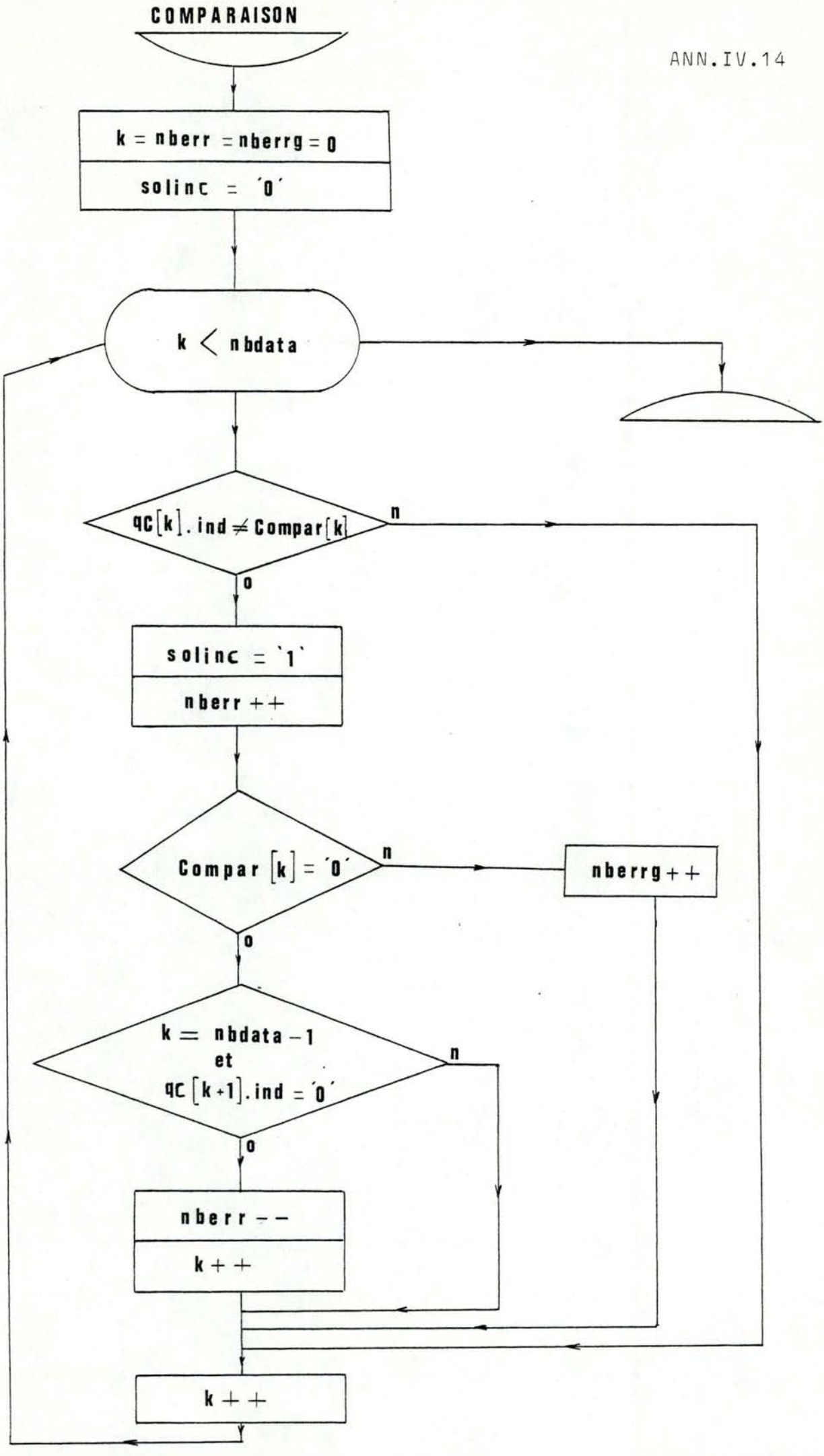
Les accumulateurs sont mis à zéro ainsi que l'écart w.tg (eca). On initialise aussi les nombres minima d'erreurs et d'erreurs graves (merr et merrg) au nombre de données (nbdata).
On positionne enfin un indicateur solpos sur '0' : on le mettra à '1' quand on aura trouvé une solution sans erreur grave (cela nous permettra que de retenir que les solutions sans erreurs graves une fois que l'on en aura trouvé une).

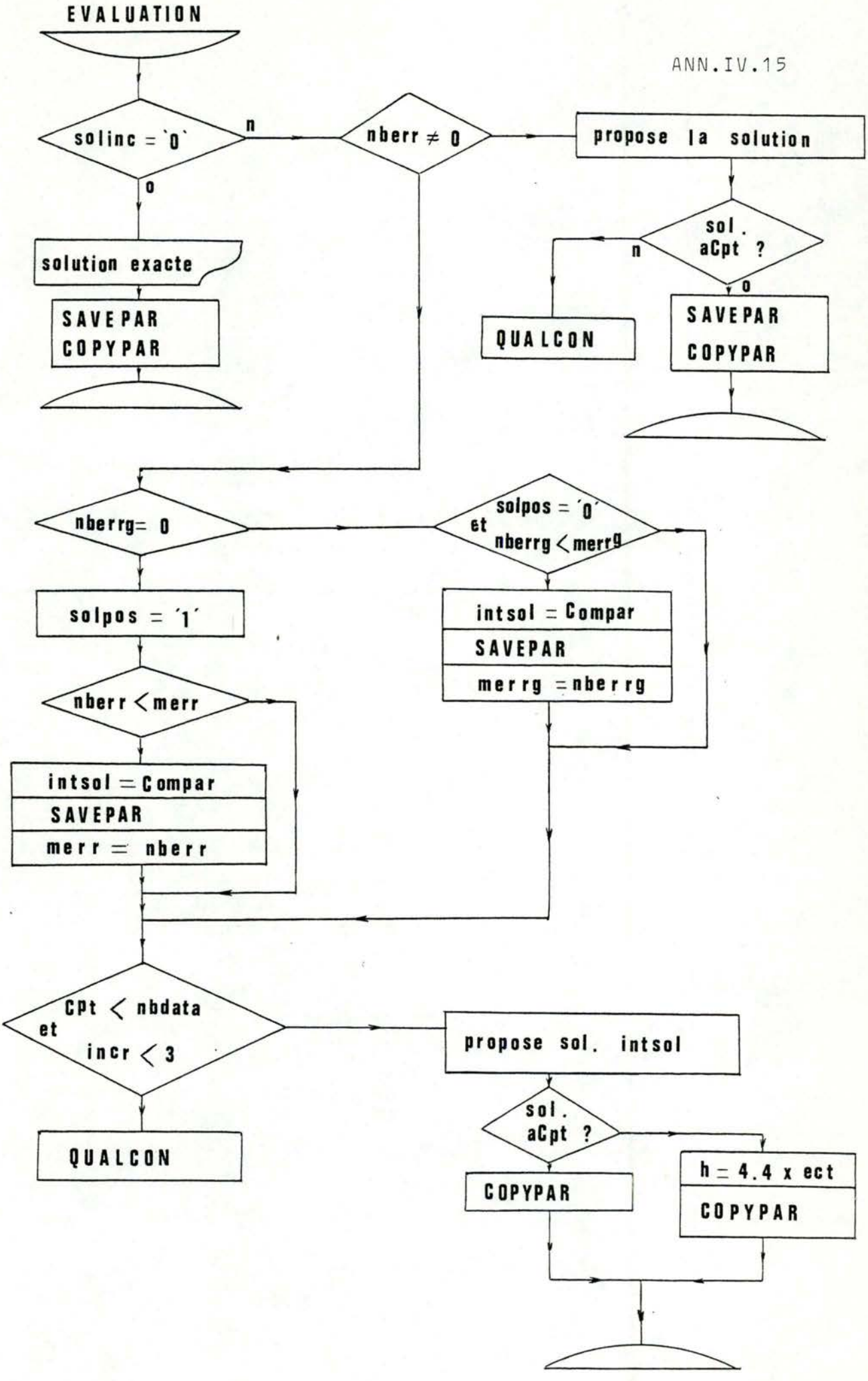
Chaque fois que le programme appellera QUALCON(), le h_{lu} sera

incrémenté de 0.1. Le h est alors calculé en multipliant h_{lu} par deux écarts-types.

On fait passer le programme de contrôle sur toutes les données que l'on a lues. On possède de ce fait les résultats dans un vecteur binaire compar. Ce vecteur est destiné à être comparé avec le but recherché. C'est ce qui sera fait dans TEST() appelé à la fin de QUALCON()


QUALCON
incr = incr + 1
Cpt = 0
h = 2 x ect x [intdeci + incr]
k = 0 ; k < nbdata ; k++
CONTROLE DE QUALITE (voir HP25)
donnee
fausse
compar k = '0'
juste
compar k = '1'
Cpt ++
TEST
TEST
COMPARAISON
EVALUATION


COMPARAISON
k = nberr = nberrg = 0
solinc = '0'
k < nbdata
qC[k].ind ≠ Compar[k]
n
o
solinc = '1'
nberr ++
Compar [k] = '0'
n
nberrg ++
o
k = nbdata - 1
et
qC [k+1].ind = '0'
n
o
nberr --
k ++
k ++


EVALUATION
solinc = '0'
n
o
nberr ≠ 0
propose la solution
sol . aCpt ?
n
o
solution exacte
SAVEPAR
COPYPAR
QUALCON
SAVEPAR
COPYPAR
nberrg= 0
solpos = '0' et nberrg < merrg
solpos = '1'
intsol = Compar
SAVEPAR
merrg = nberrg
nberr < merr
intsol = Compar
SAVEPAR
merr = nberr
CPt < nbdata et incr < 3
propose sol. intsol
sol . aCpt ?
QUALCON
COPYPAR
h = 4.4 x ect
COPYPAR

```
cat qc.c
                    /* qc.c est un programme de determination
                       en interactif des parametres de
                       controle de qualite d'apres les
                       dernieres donnees contenues dans le
                       fichier .
                    */

double sqrt();
double atof();
struct {
          float dat;
          char ind; } qc[210];
float ss,sc,moy,ect,incr,eca,intdeci,h,ac1,ac2;
char compar[210],intsol[210];
int merrs,merr;
char solpos;
int nbb;
int cpt;
int nbdata;
float par[10];
float w;
int dataf;
```

```
                        /* main contient l'ouverture du fichier,
                           la lecture des parametres et des
                           donnees ,l'initialisation des
                           variables et le premier appel
                           de qualcon .
                        */

main () {

        char *fil;
        int nrd,i,j,n;
        int npar;

    printf("filename : ");
    n = read(0,fil,10);
    fil[n-1] = '\0';
    dataf = open(fil,2);
   if(dataf == -1)  {  printf("filename not correct !\n");
                       exit();
                    }

    npar = read(dataf,par,20);

nbb = par[3];
    if (par[0] == 0)    { seek(dataf,40,0); read(dataf,ac,nbb);
                        }

        else  {  seek(dataf,-nbb,2);
                 n = read(dataf,ac,nbb);
              }

    nbdata = par[3] / 6;
printf("nbdata = #%d\n",nbdata);
                      /*  calcul de moyenne et ecart-type
                      */
    moy = par[1] / nbdata;
    ect = sqrt( par[2] / nbdata - moy * moy );

printf("  moyenne = %f\n",moy);
printf("  ecart-type = %f\n",ect);

    par[9] = moy;

                      /* initialisation  */

    incr = eca = 0;
    ac1 = ac2 = 0;
    intdeci = 1.9;
    solpos = '0';
    merr = merrs = nbdata;
    qualcon();

    close (dataf);
      }
```

```
                        /* qualcon effectue le controle de
                           qualite sur toutes les donnees lues.
                           il place les resultats trouves dans
                           un vecteur binaire "compar" et
                           appelle ensuite "test()" pour
                           evaluer la solution trouvee.
                        */

qualcon ()  {
              float a,b;
              int i;
     incr =+ 0.1;
     cpt = 0;
     h = 2 * ect * ( intdeci + incr );

     for(i=0;i<nbdata;i++)
     {
       a = ac[i].dat - moy - eca;
       b = ac[i].dat - moy + eca;

       if(ac1 >= 0)  ac1 =+a;
                   else  ac1 = a;
       if(ac1 > h)
                  {
                    ac1 = ac2 = 0;
                    compar[i] = '0';
                  }
                else
                    {
                      if(ac2 >= 0)  ac2 = b;
                                  else ac2 =+ b;
                      if(ac2 < -h)   {
                                       ac1 = ac2 = 0;
                                       compar[i] = '0';
                                     }
                                   else  { compar[i] = '1';
                                           cpt++;
                                         }
                    }
     }

     test();
}
```

```
                        /* test compare la solution trouvee
                           par qualcon avec les renseignements
                           accompagnant les donnees sur le
                           fichier .
                           selon le degre d'exactitude de la
                           solution ,il la retient ,la propose
                           ou la stocke dans "intsol".
                           il la retient si elle est exacte .
                           il la propose si nberr = 0 .
                           sinon ,il la stocke selon le principe
                           du minimum d'erreurs .
                           c'est a l'utilisateur d'accepter ou
                           de refuser une solution !
                         */

test ()  {
            int k,nberr,nberrs;
            char rep[2];
            char solinc;

     k = nberr = nberrs = 0;
     solinc = '0';

     while (k<nbdata)

     {
        if(qc[k].ind != compar[k])
        {
            solinc = '1';
            nberr++;
            if(compar[k] == '0')
                                {
                                 if (k != (nbdata - 1) && qc[k+1].ind
                                    { nberr--;k++;}                    =='0');
                                }
            else nberrs++;
           }
        k++;
     }
```

```
                        /* evaluation de la solution */

if(solinc == '0') {
                    printf("exact solution with %f \n",incr);
                    printf("the h found = %f\n",h);
                    savepar();
                    copypar();
                  }
else  {
      if(nberr == 0)

                     {  printf("proposed solution :\n");
                        printf("**********************\n\n");
                        for(k=0;k<nbdata;k++)
                             {  if ( k % 8  == 0 )

                                   printf("\n");

                                if( par[4] == 0 )

printf("%.0f %c%c  ",ac[k].dat,ac[k].ind,compar[k]);

                                  else

printf("%.2f %c%c  ",ac[k].dat,ac[k].ind,compar[k]);

                             }
                        printf("do you accept the solution ? y or n\
                                                                 n");
                 rdrep:
                        read(0,rep,2);

                        switch(rep[0]) {

                  case 'n':
                                qualcon();
                                break;
                  case 'y':
                                printf("solution found with incr=%f\n"
                                printf("the h found = %f\n",h);    ,incr);
                                savepar();
                                copypar();
                                break;
                  default:
                                printf("will you repeat ? y or n ?");
                                goto rdrep;
                                break;

                                     }

                    }
```

```
else {

if(nberrs != 0
                { if(solpos == '0' && nberrs < merrs)

                        {   for(k=0;k<nbdata;k++)
                                intsol[k] = compar[k];
                            savepar();
                            merrs = nberrs;
                        }

                }

else
    { solpos '1';
      if(nbe<merr)
                    {   for(k=0;k<nbdata;k++)
                            intsol[k] = compar[k];
                            savepar();
                            merr = nberr;
                    }
    }
                    /*   test de fin  */

      if(ct < nbdata && incr < 3)  qualcon();
      else

          {   printf("0=serious error,1=small error::%c\n",solpos);
              printf("the better h = %f\n",par[8]);
              printf("   solution =\n");
              printf("**************\n");
              for(k=0;k<nbdata;k++)
                { if( k % 8 == 0 )  printf("\n");

                    if( par[4] == 0 )

printf("%.0f %c%c  ",ac[k].dat,ac[k].ind,intsol[k]);

                     else

printf("%.2f %c%c  ",ac[k].dat,ac[k].ind,intsol[k]);
                }
              printf("do you accept it ?");
              printf(" y or n ?");
              rdrep2:
              read(0,rep,2);

                switch(rep[0]) {

                    case 'y':
                                copypar();
                                 printf("all risht");
                                break;

                    case 'n':
                                h = 2.2 * 2 * ect;
                                copypar();
                                printf('the h corresponds : ARL=3");
                                break;

                    default:
                                printf("will you repeat?y or n?");
                                goto rdrep2;
                                break;
                            }
                }
        }
    }
}
```

```
                        /*  sauvetase des parametres  */

savepar () {

                par[5] = eca;
                par[6] = ac1;
                par[7] = ac2;
                par[8] = h;
            }


                         /*  recopie des parametres en
                             debut de fichier
                         */

copypar () {

                seek(dataf,0,0);
                write(dataf,par,40);

            }
```

```
creat.o
filename = param
%
```

) création du fichier "param"

```
init.o
filename : param
is there a data ? y or n?y
donnee numero 1= 560
     caractere numero 1=1
is there a data ? y or n?y
donnee numero 2= 590
     caractere numero 2=1
is there a data ? y or n?y
donnee numero 3= 570
     caractere numero 3=1
is there a data ? y or n?y
donnee numero 4= 580
     caractere numero 4=1
is there a data ? y or n?y
donnee numero 5= 590
     caractere numero 5=1
is there a data ? y or n?y
donnee numero 6= 590
     caractere numero 6=1
is there a data ? y or n?y
donnee numero 7= 610
     caractere numero 7=1
is there a data ? y or n?y
donnee numero 8= 590
     caractere numero 8=1
is there a data ? y or n?y
donnee numero 9= 580
     caractere numero 9=0
is there a data ? y or n?y
donnee numero 10= 600
     caractere numero 10=1
is there a data ? y or n?y
donnee numero 11= 590
     caractere numero 11=1
is there a data ? y or n?y
donnee numero 12= 580
     caractere numero 12=1
is there a data ? y or n?y
donnee numero 13= 590
     caractere numero 13=1
is there a data ? y or n?y
donnee numero 14= 570
     caractere numero 14=1
is there a data ? y or n?y
donnee numero 15= 590
     caractere numero 15=1
is there a data ? y or n?y
donnee numero 16= 580
     caractere numero 16=1
is there a data ? y or n?y
donnee numero 17= 590
     caractere numero 17=1
is there a data ? y or n?y
donnee numero 18= 570
     caractere numero 18=1
is there a data ? y or n?y
```

↖ initialisation du fichier "param"

```
donnee numero 19= 580
     caractere numero 19=1
is there a data ? y or n?y
donnee numero 20= 590
     caractere numero 20=1
is there a data ? y or n?y
donnee numero 21= 570
     caractere numero 21=1
is there a data ? y or n?y
donnee numero 22= 580
     caractere numero 22=1
is there a data ? y or n?y
donnee numero 23= 570
     caractere numero 23=1
is there a data ? y or n?y
donnee numero 24= 570
     caractere numero 24=1
is there a data ? y or n?y
donnee numero 25= 590
     caractere numero 25=1
is there a data ? y or n?y
donnee numero 26= 570
     caractere numero 26=1
is there a data ? y or n?y
donnee numero 27= 560
     caractere numero 27=1
is there a data ? y or n?y
donnee numero 28= 580
     caractere numero 28=0
is there a data ? y or n?y
donnee numero 29= 580
     caractere numero 29= f
is there a data ? y or n?
donnee numero 30= 580
     caractere numero 30=1
is there a data ? y or n?
integer ? y or n ?y                n
#
```

↖ les données sont entières

qc.0

```
nbdata = #30
  moyenne = 581.333313
  ecart-type = 11.176428
Je fais qualcon
Je fais test
solinc = 1
Je fais qualcon
Je fais test
solinc = 1
proposed solution :
******************
```

points de repère dans le programme (seront supprimés par la suite).

les 2 erreurs voulues par l'utilisateur sont détectées chaque fois une donnée à l'avance

```
560 11  590 11  570 11  580 11  590 11  590 11  610 11  590 10
580 01  600 11  590 11  580 11  590 11  570 11  590 11  580 11
590 11  570 11  580 11  590 11  570 11  580 11  570 11  570 11
590 11  570 11  560 10  580 01  580 11  580 11  do you accept
m
will you repeat ? y or n ?n
Je fais qualcon
Je fais test
solinc = 1
proposed solution :
******************
```

— réponse différente de y ou n.

it ? y or n ?

solution refusée par utilisateur

```
560 11  590 11  570 11  580 11  590 11  590 11  610 11  590 10
580 01  600 11  590 11  580 11  590 11  570 11  590 11  580 11
590 11  570 11  580 11  590 11  570 11  580 11  570 11  570 11
590 11  570 11  560 10  580 01  580 11  580 11  do you accept
n
Je fais qualcon
Je fais test
solinc = 1
proposed solution :
******************
```

idem

```
560 11  590 11  570 11  580 11  590 11  590 11  610 11  590 10
580 01  600 11  590 11  580 11  590 11  570 11  590 11  580 11
590 11  570 11  580 11  590 11  570 11  580 11  570 11  570 11
590 11  570 11  560 10  580 01  580 11  580 11  do you accept
n
Je fais qualcon
Je fais test
solinc = 1
proposed solution :
******************
```

idem

```
560 11  590 11  570 11  580 11  590 11  590 11  610 11  590 10
580 01  600 11  590 11  580 11  590 11  570 11  590 11  580 11
590 11  570 11  580 11  590 11  570 11  580 11  570 11  570 11
590 11  570 11  560 10  580 01  580 11  580 11  do you accept
b
will you repeat ? y or n ?n
Je fais qualcon
Je fais test
solinc = 1
Je fais qualcon
Je fais test
solinc = 1
Je fais qualcon
```

⋮

```
Je fais test
solinc = 1
Je fais qualcon
Je fais test
solinc = 1
Je fais qualcon
Je fais test
solinc = 1
0=serious error,1=small error::0
the better h = 44.705711
   solution =
**************

560 11  590 11  570 11  580 11  590 11  590 11  610 10  590 11
580 01  600 11  590 11  580 11  590 11  570 11  590 11  580 11
590 10  570 11  580 11  590 11  570 11  580 11  570 11  570 11
590 11  570 11  560 10  580 01  580 11  580 11  do you accept
will you repeat?y or n?n
the h corresponds : ARL=3% verif.o
filename = param
        par[0] = 0.000000
        par[1] = 17440.000000
        par[2] = 10142200.000000
        par[3] = 180.000000
        par[4] = 0.000000
        par[5] = 0.000000
        par[6] = -1.333313
        par[7] = -3.999939
        par[8] = 44.705711
        par[9] = 581.333313
%
```

fin de simulation: propose la meilleure solution — contient des erreurs "graves".

solution non acceptée

? y or n ?

les paramètres sont fixés à des valeurs t.q. ARL (10) = 3

verif.o : petit programme d'impression des paramètres situés en début de fichier (utile à la vérification de la bonne marche des programmes).

Autres exemples :

```
CREAT.D
FILENAME = PARAM
% INIT.D
FILENAME : PARAM
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 1= 560
     CARACTERE NUMERO 1=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 2= 590
     CARACTERE NUMERO 2=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 3= 570
     CARACTERE NUMERO 3=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 4= 580
     CARACTERE NUMERO 4=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 5= 590
     CARACTERE NUMERO 5=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 6= 590
     CARACTERE NUMERO 6=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 7= 610
     CARACTERE NUMERO 7=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 8= 590
     CARACTERE NUMERO 8=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 9= 580
     CARACTERE NUMERO 9=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 10= 600
     CARACTERE NUMERO 10=0
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 11= 590
     CARACTERE NUMERO 11=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 12= 580
     CARACTERE NUMERO 12=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 13= 590
     CARACTERE NUMERO 13=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 14= 570
     CARACTERE NUMERO 14=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 15= 590
     CARACTERE NUMERO 15=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 16= 580
     CARACTERE NUMERO 16=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 17= 590
     CARACTERE NUMERO 17=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 18= 570
     CARACTERE NUMERO 18=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 19= 580
     CARACTERE NUMERO 19=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 20= 590
     CARACTERE NUMERO 20=1
```

) créer le fichier "param"

initialisation, avec 60 données.

```
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 21= 570
     CARACTERE NUMERO 21=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 22= 580
     CARACTERE NUMERO 22=0
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 23= 570
     CARACTERE NUMERO 23=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 24= 570
     CARACTERE NUMERO 24=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 25= 590
     CARACTERE NUMERO 25=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 26= 570
     CARACTERE NUMERO 26=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 27= 560
     CARACTERE NUMERO 27=0
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 28= 580
     CARACTERE NUMERO 28=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 29= 590
     CARACTERE NUMERO 29=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 30= 590
     CARACTERE NUMERO 30=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 31= 580
     CARACTERE NUMERO 31=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 32= 580
     CARACTERE NUMERO 32=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 33= 590
     CARACTERE NUMERO 33=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 34= 580
     CARACTERE NUMERO 34=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 35= 570
     CARACTERE NUMERO 35=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 36= 580
     CARACTERE NUMERO 36=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 37= 590
     CARACTERE NUMERO 37=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 38= 570
     CARACTERE NUMERO 38=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 39= 580
     CARACTERE NUMERO 39=0
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 40= 570
     CARACTERE NUMERO 40=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 41= 580
     CARACTERE NUMERO 41=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 42= 580
     CARACTERE NUMERO 42=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 43= 590
     CARACTERE NUMERO 43=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 44= 580
     CARACTERE NUMERO 44=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 45= 580
     CARACTERE NUMERO 45=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 46= 580
     CARACTERE NUMERO 46=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 47= 590
     CARACTERE NUMERO 47=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 48= 580
     CARACTERE NUMERO 48=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 49= 590
     CARACTERE NUMERO 49=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 50= 570
     CARACTERE NUMERO 50=0
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 51= 590
     CARACTERE NUMERO 51=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 52= 580
     CARACTERE NUMERO 52=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 53= 570
     CARACTERE NUMERO 53=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 54= 570
     CARACTERE NUMERO 54=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 55= 580
     CARACTERE NUMERO 55=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 56= 580
     CARACTERE NUMERO 56=0
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 57= 590
     CARACTERE NUMERO 57=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 58= 580
     CARACTERE NUMERO 58=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 59= 590
     CARACTERE NUMERO 59=1
IS THERE A DATA ? Y OR N?Y
DONNEE NUMERO 60= 580
     CARACTERE NUMERO 60=1
IS THERE A DATA ? Y OR N?N
INTEGER ? Y OR N ?Y
%
```

données entières

```
  QC.O
FILENAME : PARAM
NBDATA = #60
  MOYENNE = 581.333313
  ECART-TYPE = 9.393928
O=SERIOUS ERROR,1=SMALL ERROR::O
THE BETTER H = 39.454494
   SOLUTION =
**************

560 11  590 11  570 11  580 10  590 11  590 11  610 10  590 11
580 11  600 01  590 11  580 11  590 10  570 11  590 11  580 11
590 11  570 11  580 11  590 11  570 11  580 01  570 11  570 11
590 11  570 10  560 01  580 11  590 11  590 11  580 11  580 11
590 11  580 11  570 11  580 11  590 11  570 11  580 01  570 11
580 11  580 11  590 11  580 11  580 11  580 11  590 11  580 11
590 11  570 01  590 11  580 11  570 11  570 11  580 11  580 00
590 11  580 11  590 11  580 11  DO YOU ACCEPT IT ? Y OR N ?Y
ALL RIGHT%
```

pas de solution "proposed"
=> solution finale

solution acceptée !

on ajoute 25 données en les contrôlant avec les paramètres que l'on vient de déterminer.

```
% AJOUT.O
 FILENAME : PARAM
 THE DATA IS : 580
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 580
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 570
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 590
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 580
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 590
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 590
1 = BON , 0 = INCORRECT  :  0
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 580
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 570
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 590
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 590
1 = BON , 0 = INCORRECT  :  0
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 590
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
```

```
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  0
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 580
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  0
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 590
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 580
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 590
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?N
%
```

on recalcule les paramètres avec 85 données au lieu de 60.

```
  QC,0
FILENAME : PARAM
NBDATA = #85
  MOYENNE = 583.764709
  ECART-TYPE = 10.172013
0=SERIOUS ERROR,1=SMALL ERROR::0
THE BETTER H = 42.722454
   SOLUTION =
***************

560 11  590 11  570 11  580 11  590 11  590 11  610 11  590 10
580 11  600 01  590 11  580 11  590 11  570 11  590 11  580 11
590 11  570 11  580 11  590 11  570 11  580 01  570 10  570 11
590 11  570 11  560 00  580 11  590 11  590 11  580 11  580 11
590 11  580 11  570 11  580 11  590 11  570 11  580 01  570 10
580 11  580 11  590 11  580 11  580 11  580 11  590 11  580 11
590 11  570 01  590 11  580 11  570 11  570 11  580 10  580 01
590 11  580 11  590 11  580 11  580 11  580 11  570 11  590 11
580 11  590 11  600 11  590 01  600 10  580 11  600 11  570 11
590 11  590 01  600 11  590 11  600 00  580 11  600 11  600 11
600 00  590 11  580 11  590 11  600 11  DO YOU ACCEPT IT ? Y
ALL RIGHT%
```

on se rapproche de plus en plus de la sol. vraie (solution acceptée)

on ajoute encore quelques données.

```
AJOUT.O
FILENAME : PARAM
 THE DATA IS : 580
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 590
1 = BON , 0 = INCORRECT  :  0
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 580
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 580
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 580
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 590
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 600
1 = BON , 0 = INCORRECT  :  0
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 590
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?Y
 THE DATA IS : 580
1 = BON , 0 = INCORRECT  :  1
DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
IS THERE ANOTHER DATA  ? Y OR N ?600
% 0: NOT FOUND
```

réponse ≠ y considérée comme = n !

on est obligé de relancer le programme.

```
% AJOUT.O
  FILENAME : PARAM
  THE DATA IS : 600
 1 = BON , 0 = INCORRECT  :  1
 DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
 IS THERE ANOTHER DATA  ? Y OR N ?Y
  THE DATA IS : 590
 1 = BON , 0 = INCORRECT  :  1
 DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
 IS THERE ANOTHER DATA  ? Y OR N ?Y
  THE DATA IS : 590
 1 = BON , 0 = INCORRECT  :  1
 DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
 IS THERE ANOTHER DATA  ? Y OR N ?Y
  THE DATA IS : 590
 1 = BON , 0 = INCORRECT  :  1
 DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
 IS THERE ANOTHER DATA  ? Y OR N ?Y
  THE DATA IS : 600
 1 = BON , 0 = INCORRECT  :  0
 DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
 IS THERE ANOTHER DATA  ? Y OR N ?Y
  THE DATA IS : 590
 1 = BON , 0 = INCORRECT  :  1
 DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
 IS THERE ANOTHER DATA  ? Y OR N ?Y
  THE DATA IS : 580
 1 = BON , 0 = INCORRECT  :  1
 DO YOU ACCEPT THE SOLUTION ? Y OR N ?Y
 IS THERE ANOTHER DATA  ? Y OR N ?N
%
```

détermination des paramètres avec 104 données !

```
  QC.O
FILENAME : PARAM
NBDATA = #104
  MOYENNE = 584.903870
  ECART-TYPE = 10.093878
0=SERIOUS ERROR,1=SMALL ERROR:?0
THE BETTER H = 42.394287
   SOLUTION =
****************
```

sol. finale

↖ erreurs "graves"

```
560 11  590 11  570 11  580 10  590 11  590 11  610 11  590 11
580 11  600 00  590 11  580 11  590 11  570 11  590 11  580 11
590 11  570 11  580 11  590 11  570 11  580 00  570 11  570 11
590 11  570 11  560 00  580 11  590 11  590 11  580 11  580 11
590 11  580 11  570 11  580 11  590 11  570 11  580 00  570 11
580 11  580 11  590 11  580 11  580 11  580 11  590 11  580 11
590 11  570 00  590 11  580 11  570 11  570 11  580 11  580 00
590 11  580 11  590 11  580 11  580 11  580 11  570 11  590 11
580 11  590 11  600 11  590 01  600 11  580 11  600 10  570 11
590 11  590 01  600 11  590 11  600 00  580 11  600 11  600 11
600 00  590 11  580 11  590 11  600 11  580 11  600 11  590 01
600 10  580 11  580 11  600 11  580 11  590 11  600 01  590 11
580 11  600 10  590 11  590 11  590 11  600 01  590 11  580 11
DO YOU ACCEPT IT ? Y
ALL RIGHT%
```

↖ solution acceptée !

```
          VERIF.O
FILENAME = PARAM
      PAR[0] = 0.000000
      PAR[1] = 60830.000000
      PAR[2] = 35590300.000000
      PAR[3] = 624.000000
      PAR[4] = 0.000000
      PAR[5] = 0.000000
      PAR[6] = 30.576782
      PAR[7] = -4.903870
      PAR[8] = 42.394287
      PAR[9] = 584.903870
%
```

Paramètres de début de fichier.