CHAPITRE IV

LES ALGORITHMES D'ACCELERATION DE LA CONVERGENCE DES SUITES.

IV-1. Convergence des suites.

Jusqu'à présent, nous n'avons considéré que l'accélération de la convergence de suites un peu particulières, du type :

$$f(z) = \sum_{0}^{\infty} c_n x^n \qquad \text{(IV-1)}$$

en considérant qu'une telle série peut être étudiée comme suite de ses sommes partielles. Nous envisageons, à présent, la convergence des suites quelconques.
Considérons la suite $\{S_n\} = S_0, S_1, S_2, \ldots$ qui converge vers la limite S. Les questions essentielles qui se posent sont les suivantes :

- Comment estimer quantitativement la vitesse de convergence de cette suite ?
- S'il s'avère que cette vitesse est faible, est-il possible de l'augmenter en remplaçant la suite $\{S_n\}$ par une autre, $\{\sigma_n\}$ par exemple, qui tendrait, plus rapidement, vers la même limite?

Une réponse satisfaisante à la deuxième question est loin d'être évidente. Nous allons présenter un grand nombre d'algorithmes qui poursuivent ce but. Leur nombre indique qu'aucun algorithme universel n'existe qui serait valable pour tous les cas : cela signifie qu'il y a lieu d'étudier chaque cas séparément afin de dégager les règles qui permettent d'opérer le choix de l'algorithme optimal. Cette situation était déjà illustrée par les approximants de Padé qui accéléraient la série exponentielle mais échouaient complètement en face de la série dilogarithme.

Dans le but d'illustrer au maximum les algorithmes très variés qui vont être présentés, nous avons sélectionné quelques séries "test" destinées à subir ces divers algorithmes. Nous les définissons une fois pour toutes et nous nous y référerons systématiquement en cours d'exposé. Nous avons choisi ces suites afin d'illustrer le plus grand nombre de cas susceptibles de se présenter en pratique. Elles sont extraites du livre de Wimp (Sequence transformations and their applications.- Academic Press).

Suite S_n	Limite S (asymptote principale)
$(LN2)_n = \sum_0^n \frac{(-1)^k}{k+1}$	ln 2 = 0.693147180559945... (1/n)
$(RC)_n = \sum_0^n \frac{(-1)^k}{\sqrt{k+1}}$	0.604898643421630... (1/√n)
$(PIC)_n = \sum_0^n \frac{1}{(k+1)^2}$	$\pi^2/6$ = 1.644934066846559... (1/n)
$(FAC)_n = \sum_0^n (-1)^k k!$	divergente, prolongée par $\int_0^\infty \frac{e^{-t}}{t+1} dt = 0.5963473611...$
$(FOU)_n = \sum_0^n \frac{\sin(k)}{k}$	$\frac{\pi - 1}{2} = 1.070796327...$
$(IT1)_n$ générée par $S_{n+1} = \frac{20}{S_n^2 + 2S_n + 10}$ $(S_0 = 1)$	1.368808107...
$(IT2)_n$ générée par $S_{n+1} = \frac{20 - 2S_n^2 - S_n^3}{10}$ $(S_0 = 1)$	divergente
$(LUB)_n = \sum_0^n \frac{(-1)^{\left\lceil \frac{k}{2} \right\rceil}}{k+1}$	1.131971754

Ces suites ont été choisies suffisamment lentes, au départ, pour exiger un procédé d'accélération. Pour quantifier la vitesse de convergence d'une suite, il suffit d'étudier son comportement asymptotique. On peut écrire, dans tous les cas :

$S_n = S + as(n)$

en sorte que l'erreur relative que l'on commet sur S en se limitant à S_n vaut :

$$e^{-p} \approx \left| \frac{S - S_n}{S} \right| = \left| \frac{as(n)}{S} \right|$$

La connaissance du comportement asymptotique de la suite permet donc d'en estimer la convergence. En général, as(n) n'est connue qu'à une constante multiplicative près, en sorte que l'on se borne à écrire, $p = -\ln |as(n)|$, où on a négligé le logarithme de la constante S qui est généralement petit. p indique le nombre de chiffres népériens corrects dans S_n , soit 2.3

fois le nombre de chiffres décimaux. Pour fixer les idées, nous appellerons p la précision. En fin de chapitre IV, au terme de l'étude des divers procédés d'accélération, nous leur soumettrons l'ensemble des suites "test" et nous dresserons le tableau comparatif, IV-4, des précisions avant et après transformation.

Il n'existe malheureusement pas de règle absolument générale pour obtenir l'asymptote as(n) par voie analytique. Lorsque la suite est composée des approximants d'une fraction continue, nous avons vu au chapitre II comment le théorème de Pincherle permet d'obtenir une expression pour as(n) : il suffit d'estimer le contraste entre les solutions dominantes et dominées de la récurrence associée. Pour ce qui est des séries, le calcul de as(n) est déjà moins simple. On trouvera dans l'ouvrage de De Bruijn (Asymptotic methods in Analysis - North Holland) les bases de la théorie sous-jacente. Une procédure simple consiste à essayer la formule sommatoire de MacLaurin :

$$\sum_{k=1}^{n} f_k \approx \int_0^n f(k)dk - \frac{1}{2}[f(0)+f(n)] + \frac{1}{12}[f'(n)-f'(0)] - \frac{1}{720}[f'''(n)-f'''(0)] + \ldots$$

Exemple : Soit la suite : $S_n = (PIC)_n = \sum_0^n \frac{1}{(k+1)^2} = \sum_1^{n+1} \frac{1}{k^2}$

Il est clair que $f(k) = 1/k^2$, que ses dérivées tendent asymptotiquement vers 0 et que sa primitive fournit le terme dominant. On est donc certain du comportement asymptotique suivant :
$(PIC)_n \approx S + \lambda/n$ où λ est une constante inconnue et S vaut $\pi^2/6$.
On voit que cette suite converge lentement puisque 10^6 termes garantissent à peine 6 chiffres exacts. Il y a lieu de remarquer sur cet exemple que la formule de Mac Laurin permet de raffiner le développement asymptotique. On peut, en effet, écrire :

$(PIC)_n \approx S + \lambda_1/n + \lambda_2/n^2 + \lambda_3 n^3 + \ldots$ où les constantes λ_i sont inconnues.

On voit ainsi apparaître le concept d'échelle asymptotique :

$S_n \approx S + \lambda_1 as^{(1)}(n) + \lambda_2 as^{(2)}(n) + \lambda_3 as^{(3)}(n) + \ldots$ où $as^{(1)}(n)$ est l'asymptote dominante et $as^{(i)}(n)$, (i>1), sont les asymptotes dominées.

On a, de plus, que : $\lim_{n\to\infty} [as^{(j)}(n)/ as^{(i)}(n)] = 0$ si $j > i$.

On verra plus loin (algorithme E) que la connaissance de l'échelle asymptotique complète permet, en quelque sorte, d'optimaliser l'accélération de la convergence de la suite. Malheureusement, elle n'est pas toujours totalement accessible et il faut souvent se contenter de $as^{(1)}(n)$. On notera que l'échelle asymptotique ne fixe pas, de façon univoque, la valeur numérique des coefficients λ_i du fait que toute combinaison linéaire d'asymptotes distinctes, d'ordre i et plus, peut être considérée elle-même comme une asymptote d'ordre i. Cela n'hypothèque heureusement en rien les possibilités d'accélération de la convergence de la suite S_n .

IV-2. Les procédés de sommation linéaire.

Initialement conçus pour attribuer une limite aux séries divergentes (cfr Hardy, Divergent series, Oxford Press) par une sorte de prolongement analytique, les procédés de sommation peuvent être appliqués aux séries lentement convergentes dans le but de les accélérer. La classe la plus importante des procédés de sommation linéaire est basée sur l'application d'une matrice triangulaire $A = (a_{ij})$ qui transforme la suite originale, S_n, en une suite modifiée, T_n, définie comme suit :

$$\begin{pmatrix} T_0 \\ T_1 \\ T_2 \\ \vdots \end{pmatrix} = A \begin{pmatrix} S_0 \\ S_1 \\ S_2 \\ \vdots \end{pmatrix}$$

$$avec \qquad A = \begin{pmatrix} a_{00} & & & \\ a_{10} & a_{11} & & 0 \\ a_{20} & a_{21} & a_{22} & \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

Appelons A ce procédé de sommation, du nom de la matrice qui le symbolise. On dit qu'il est régulier si, la suite S_n convergeant vers S, la suite transformée, T_n, converge vers la même limite. Un théorème dû à Toepliz permet d'affirmer la régularité du procédé A sous trois hypothèses :

$$(i) \qquad \sum_{k=0}^{\infty} |a_{nk}| < M \qquad \forall n$$

$$(ii) \qquad \lim_{n\to\infty} a_{nk} = 0 \qquad \forall k$$

$$(iii) \qquad \lim_{n\to\infty} \sum_{k=0}^{\infty} a_{nk} = 1$$

Ces conditions sont suffisantes sans être nécessaires car il existe des procédés qui y dérogent et qui sont cependant réguliers pour certaines familles de suites. En fait, le théorème de Toepliz s'applique également lorsque la matrice A n'est pas triangulaire, mais en pratique, c'est toujours le cas triangulaire qui est utilisé.

Les principaux procédés réguliers étudiés sont :

1) Le procédé (H,1) de Hölder. La matrice A est donnée par :

$a_{ij} = 1/(i+1)$ pour j=1, ..., i.
$=0$ pour j>i

Un procédé plus évolué, noté (H,K), utilise la matrice A^K (K entier).

Concrètement, on écrit :

$$(H,1)\equiv\begin{pmatrix} 1/1 & 0 & 0 & 0 \\ 1/2 & 1/2 & 0 & 0 \\ 1/3 & 1/3 & 1/3 & 0 \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

d'où la suite transformée, T_n , des moyennes arithmétiques :

$$T_0 = S_0$$
$$T_1 = \frac{S_0 + S_1}{2}$$
$$T_2 = \frac{S_0 + S_1 + S_2}{3}$$
$$\vdots$$

Ensuite, on définit la matrice (H,2) par la relation :

$$(H,2) = (H,1) * (H,1)$$

d'où la nouvelle suite U_n :

$$U_0 = T_0$$
$$U_1 = \frac{T_0 + T_1}{2}$$
$$U_2 = \frac{T_0 + T_1 + T_2}{3}$$

Exemples : soit les deux suites divergentes oscillantes :

a) S_n={1, 0, 1, 0, 1, ...}, d'expression générale : $S_n = \sum_0^n (-1)^k = \frac{1+(-1)^n}{2}$

On trouve, facilement : T_n={1, 1/2, 2/3, 2/4, 3/5, 3/6, ...}$\rightarrow$1/2

b) S_n={1, -1, 2, -2, 3, ...}, d'expression générale : $S_n = \sum_0^n (-1)^k (k+1) = \frac{1+(-1)^n (2n+3)}{4}$

On trouve successivement : T_n={1, 0, 2/3, 0, 3/5, 0, 4/7, 0, ...}$\equiv \frac{1}{4}\frac{n+2}{n+1}\left[1+(-1)^n\right]$

U_n={1, 1/2, 5/9, 5/12, 34/75, 34/90, ...} $\rightarrow$1/4

On peut légitimement s'interroger sur le sens à donner à la limite d'une série divergente. Les limites des suites prises en exemple sont en fait données par le prolongement analytique des fonctions $\sum_0^\infty z^k$ et $\sum_0^\infty (k+1)z^k$ en z=-1. On pourrait penser que les fonctions $(1-z)^{-1}$ et $\sum_0^\infty z^k$ contiennent la même quantité d'information. Cependant $(1-z)^{-1}$ est définie partout, sauf en z=1, tandis que $\sum_0^\infty z^k$ n'est définie, dans le plan complexe, que dans le cercle $|z|<1$ et l'idée du procédé de sommation est précisément de retrouver l'information perdue.

Le deuxième exemple concerne, similairement, la fonction $(1-z)^{-2}$ et la série équivalente $\sum_0^\infty (k+1)z^k$.

2) Le procédé de Cesaro d'ordre K (C,K).

Sa matrice d'ordre K, notée symboliquement (C,K), vaut: $(C,K)=H(1)*L^{K-1}$,
où la matrice L est triangulaire inférieure gauche, avec tous éléments égaux à l'unité.

3) Le procédé d'Euler.

Les éléments de la matrice A, du procédé d'Euler, sont donnés par :

$$a_{ij} = \frac{q^{i-j}}{(q+1)^i} C_i^j \qquad pour \quad j = 0,1,...,i$$

$$= 0 \qquad pour \quad j > i$$

$$\left(\sum_{j=0}^{i} a_{ij} = 1 \right)$$

où le paramètre q peut être choisi arbitrairement (souvent on prend q=1).

Il est facile de voir que ces trois procédés sont réguliers. Ils sont même totaux, c'est-à-dire que l'on a en plus que $\sum_{k=0}^{n} a_{nk} = 1 \quad \forall n$, ce qui garantit automatiquement la troisième hypothèse du théorème de Toepliz.

On peut présenter de deux autres manières la théorie des procédés de sommation.

IV-2.1. Moyennes de Nörlund.

Une première manière consiste à définir la suite transformée sous la forme d'une moyenne pondérée des S_n :

$$t_n = \frac{p_n S_0 + p_{n-1} S_1 + \cdots + p_0 S_n}{p_0 + \cdots + p_n}$$

Le procédé est régulier si $\frac{p_i}{\sum p_k} \to 0 \qquad \forall i$.

Le procédé (C,K) de Césaro rentre dans cette catégorie avec le choix pondéral suivant :

$$p_m = \frac{\Gamma(m+K)}{\Gamma(m+1)\Gamma(K)}$$

De même, le procédé d'Euler correspond au choix pondéral : $p_m = q^m C_n^m$.

Par contre les procédés d'Hölder, (H,K), y échappent sauf évidemment (H,1)≡(C,1). La théorie des moyennes d'Hausdorff est cependant plus générale en sorte que nous préférons nous attarder sur celle-ci.

IV-2.2. Moyennes d'Hausdorff.

La suite transformée s'écrit sous la forme :

$$t_k^{(0)} = \sum_{m=0}^{k} (-1)^{k-m} C_k^m \left(\Delta^{k-m} \mu_m \right) \; S_m$$

où μ_m est une suite auxiliaire (telle que $\mu_0 = 1$), choisie en sorte que le procédé qui en résulte soit régulier. On peut adopter un point de vue encore plus général en appliquant le procédé à la suite S considérée comme démarrant à un rang n quelconque, ce qui donne la forme canonique du procédé d'Hausdorff :

$$t_k^{(n)} = \sum_{m=0}^{k} (-1)^{k-m} C_k^m (\Delta^{k-m} \mu_m) S_{n+m} \qquad \text{(IV-2)}$$

Tous calculs faits, on dispose les résultats dans un tableau à double entrée de ce type :

$$\begin{array}{llll} t_0^{(0)} = S_0 & t_1^{(0)} & t_2^{(0)} & \cdots \\ t_0^{(1)} = S_1 & t_1^{(1)} & t_2^{(1)} & \cdots \\ t_0^{(2)} = S_2 & t_1^{(2)} & t_2^{(2)} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{array}$$

Dans cette table, la première colonne reproduit la suite de départ car $\mu_0=1$. La deuxième colonne livre le résultat de la première étape de l'algorithme et ainsi de suite en suivant les colonnes successives qui accélèrent éventuellement la convergence de la suite de départ. On trouvera dans l'ouvrage de Hardy, déjà cité, les conditions que doit remplir la suite μ_n afin que le procédé de sommation soit régulier.

Reprenons les trois exemples déjà étudiés :

- procédé (H, K) de Hölder : $\mu_m = \dfrac{1}{(m+1)^K}$

- procédé (C,K) de Cesaro : $\mu_m = \dfrac{1}{C_{m+K}^K}$

- procédé d'Euler : $\mu_m = \dfrac{1}{(q+1)^m}$

L'application de ces procédés exige le calcul des différences successives de la suite μ_m. Ce calcul est trop complexe dans le cas Hölder pour être envisagé par voie analytique. Par contre, il est faisable dans les deux autres cas. On trouve successivement :

- cas Césaro (C,K) : $\Delta^p \mu_n = \dfrac{K!\,n!}{(n+K+p)!} \dfrac{(K+p-1)!}{(K-1)!} (-1)^p$

d'où : $t_k^{(n)} = \dfrac{\sum_{m=0}^{k} C_{K+k-m-1}^{K-1} S_{n+m}}{C_{K+k}^K}$ (IV-3)

- cas Euler : $\Delta^p \mu_n = (-1)^p \dfrac{q^p}{(q+1)^{p+n}}$

d'où : $t_k^{(n)} = \sum_{m=0}^{k} C_k^m \dfrac{q^{k-m}}{(q+1)^k} S_{n+m}$ (IV-4)

Les formules (III-3) et (III-4), faciles à programmer, sont peu utilisées en pratique à cause de leurs performances modestes. Leur étude théorique n'est cependant pas dénuée d'intérêt dans la mesure où elle permet de comprendre pourquoi les méthodes linéaires accélèrent rarement les suites de façon spectaculaire. Le diagnostic d'impuissance étant posé, il devient alors possible de construire d'autres procédés, généralement non linéaires, qui évitent tous les inconvénients.

IV-3. Noyau d'un procédé de sommation.

Le formalisme d'Hausdorff est particulièrement adapté à la détermination du noyau des procédés de sommation linéaire. On appelle noyau d'un procédé de sommation l'ensemble des suites que ce procédé transforme en la suite constante {S} à une étape, k, quelconque mais fixée de son développement. On a donc $t_k^{(n)} = S$ pour tout n et pour k fixé.

L'algorithme est donc exact pour les suites qui font partie de son noyau. On conçoit que, plus le noyau d'un procédé est riche, plus ce procédé risque d'être intéressant en ce qu'il accélérera une classe plus étendue de suites.

La détermination du noyau d'Hausdorff nécessite le lemme suivant :

Lemme : Quelle que soit la suite μ_n , on a l'identité : $\sum_{m=0}^{k}(-1)^{k-m}C_k^m(\Delta^{k-m}\mu_{n+m}) = \mu_n$

Preuve : considérons l'opérateur $(E - \Delta)^k$ qui se réduit par définition à l'identité.

Il s'écrit encore : $(E-\Delta)^k = \sum_{m=0}^{k}(-1)^{k-m}C_m^k E^m \Delta^{k-m} \equiv 1$

Il suffit d'appliquer cette identité à la suite μ_n pour trouver le résultat annoncé.

Les suites S_n qui font partie du noyau du procédé d'Hausdorff sont telles qu'à une certaine étape k, on a : $t_k^{(n)} = S \qquad \forall n$

On a donc, se reportant à (IV-2) et utilisant le résultat du lemme :

$$\sum_{m=0}^{k}(-1)^{k-m}C_k^m(\Delta^{k-m}\mu_m)S_{n+m} = S$$

$$= S\sum_{m=0}^{k}(-1)^{k-m}C_k^m(\Delta^{k-m}\mu_m)$$

soit encore : $\sum_{m=0}^{k}(-1)^{k-m}C_k^m(\Delta^{k-m}\mu_m)(S_{n+m} - S) = 0$

Ceci n'est rien d'autre qu'une récurrence à coefficients constants, satisfaite par la suite $S_n - S$. Sa solution générale s'écrit donc, à chaque étape k, dans le cas probable de racines distinctes de l'équation caractéristique :

$$S_n = S + \lambda_1 z_1^n + \cdots + \lambda_k z_k^n$$

Le noyau des procédés d'Hausdorff est donc constitué de combinaisons **quelconques** d'exponentielles (éventuellement flanquées d'un polynôme si une ou plusieurs racines sont multiples) dont les arguments z_i sont **imposés** par le procédé. Les z_i changent d'ailleurs de valeur à chaque étape k.

On conçoit que l'imposition des valeurs des z_i est une restriction sévère qui rend le noyau tellement ténu que le procédé y perd une bonne partie de son utilité. On comparera ultérieurement avec l'algorithme ε qui lève cette restriction.

Exemple 1 : considérons le procédé (C, 1). On a :

$$t_k^{(n)} = \frac{\sum_{m=0}^{k} C_{k-m}^{0} S_{n+m}}{C_{k+1}^{1}} = \frac{\sum_{m=0}^{k} S_{n+m}}{k+1}$$

Si $t_k^{(n)} = S$, pour tout n, k fixé, on voit que $S_{n+m} - S$ obéit à une récurrence à coefficients constants d'équation caractéristique :

$$z^k + z^{k-1} + \cdots + z + 1 = 0 = \frac{z^{k+1} - 1}{z - 1}$$

d'où il résulte que les suites, qui appartiennent au noyau de (C, 1), sont de la forme :

$$S_n = S + \lambda_1 z_1^n + \cdots + \lambda_k z_k^n \qquad \text{(IV-5)}$$

où les z_i sont les racines $(k+1)^{èmes}$ de l'unité (1 étant exclu) et où les λ_i sont quelconques (réels ou complexes).

Exemple 2 : considérons le procédé d'Euler.
Procédant de même, on trouve l'équation caractéristique :

$\sum_{m=0}^{k} C_k^m q^{k-m} z^m = 0$ qui se réduit à: $(1 + z/q)^k = 0$, dont la racine est -q de multiplicité k.

Les suites faisant partie du procédé d'Euler, dans sa $k^{ème}$ étape, sont donc de la forme :

$$S_n = S + (-q)^n P_{k-1}(n)$$

(IV-6)

où $P_{k-1}(n)$ désigne un polynôme quelconque de degré k - 1 de la variable n.

En résumé, le procédé d'Euler somme exactement les suites S_n exponentielles polynômes de type (IV-6) tandis que le procédé de Cesaro somme exactement les combinaisons linéaires d'exponentielles de type (IV-5) d'arguments imposés.

IV-4. Autres procédés de sommation linéaire.

Tous les procédés linéaires ne sont pas inintéressants même si leur noyau est relativement plus pauvre que celui de leurs homologues non linéaires. Assez curieusement, les plus intéressants sont en fait, non réguliers, ce qui ne les empêche pas de préserver la limite d'un certain nombre de suites. On ne les utilise donc que pour ces suites-là. Ils s'écrivent sous la forme d'une combinaison linéaire, à coefficients variables avec n, des termes de la suite S_{n+m} .
Un exemple utile est fourni par le procédé de Salzer :

$$t_k^{(n)} = \sum_{m=0}^{k} \frac{(-1)^{m+k}}{k!} (n+m+\gamma)^k C_k^m S_{n+m}$$

où γ est un paramètre dont la valeur n'a guère d'importance, on pose généralement $\gamma = 1$.
Il tire son intérêt du fait que ses coefficients a_{km} dépendent de n; cela a pour conséquence que l'équation récurrente qui définit le noyau n'est plus à coefficients constants. Les suites qui font partie du noyau ne se réduisent plus dès lors à la seule classe des exponentielles polynômes comme dans les cas Cesaro et Euler. Il est d'ailleurs facile de voir que la $k^{ème}$ étape de cet algorithme possède comme noyau l'ensemble des suites :

$$S_n = S + \sum_{i=1}^{k} \lambda_i /(n+\gamma)^i$$

en sorte que l'application répétée de l'algorithme doit permettre d'accélérer les suites se comportant comme :

$$S_n \approx S + \frac{\lambda_1}{n+\gamma} + \frac{\lambda_2}{(n+\gamma)^2} + \cdots$$

Tel est le cas, par exemple, de la suite-test notée $(PIC)_n$.
Nous n'insistons pas davantage ici car ce type de procédé apparaîtra plus loin comme un cas particulier de l'algorithme E.

IV-5.1. Algorithmes deltoïdes linéaires.

Les algorithmes de triangle du type : $t_k^{(n)} = \sum_{m=0}^{k} a_{km} S_{n+m}$ peuvent, dans certains cas, être avantageusement reformulés en termes d'algorithmes deltoïdes qui présentent l'avantage d'être récursifs. L'un d'eux est bien connu : il s'agit du procédé d'extrapolation de Richardson :

$$\begin{cases} t_{k+1}^{(n)} = \dfrac{x_n t_k^{(n+1)} - x_{n+k+1} t_k^{(n)}}{x_n - x_{n+k+1}} \\ t_0^{(n)} = S_n \end{cases} \qquad (n,k = 0,1,...)$$

où la suite auxiliaire x_n tend vers 0.

Il est connu que $t_k^{(n)}$ est la valeur en x = 0 du polynôme d'interpolation de degré k qui passe par les k+1 couples : $(x_n , S_n), (x_{n+1} , S_{n+1}), \ldots ,(x_{n+k} , S_{n+k})$.

Il existe une relation entre la matrice (a_{km}) et la suite x_n :

$$a_{km} = \prod_{i=0, i\neq m}^{k} \frac{x_{n+i}}{x_{n+i} - x_{n+m}} \qquad avec \qquad \sum_{m=0}^{k} a_{km} = 1$$

Exemple : Le choix $x_i = 1/(i+1)$ mène aux éléments de matrice :

$$a_{km} = \frac{(-1)^{m+k}}{k!}(n+m+1)^k C_k^m$$

qui correspondent au procédé de Salzer avec le choix standard $\gamma = 1$. Celui-ci admet donc la formulation récursive suivante :

$$\begin{cases} t_{k+1}^{(n)} = \dfrac{(n+k+2)t_k^{(n+1)} - (n+1)t_k^{(n)}}{k+1} \\ t_0^{(n)} = S_n \qquad\qquad (n,k \geq 0) \end{cases}$$

Rappelons que le procédé de Salzer n'est pas régulier, ce qui ne l'empêche pas d'être très efficace sur un ensemble restreint de suites.
La forme générale des algorithmes deltoïdes est la suivante :

$$\begin{cases} t_{k+1}^{(n)} = \alpha_k t_k^{(n+1)} + \beta_k t_k^{(n)} \\ t_0^{(n)} = S_n \qquad\qquad n,k \geq 0 \end{cases} \qquad \text{(IV-7)}$$

La condition $\alpha_k + \beta_k = 1$ est suffisante pour assurer la régularité de l'algorithme. Le procédé de Salzer n'y obéit pas. Les procédés de sommation réguliers sont caractérisés par une matrice triangulaire d'éléments a_{ij} tels que :

$$a_{k+1,m} = \alpha_k a_{k,m-1} + (1-\alpha_k) a_{k,m}$$

Cette relation s'obtient de suite en posant $t_k^{(n)} = \sum_{m=0}^{k} a_{k,m} S_{n+m}$ dans l'algorithme (IV-7).

Elle s'inverse en :

$$\alpha_k = \frac{a_{k+1,m} - a_{k,m}}{a_{k,m-1} - a_{k,m}} \qquad \text{(IV-8)}$$

α_k doit être indépendant de m afin d'assurer la régularité de l'algorithme deltoïde.

Exemple : La méthode de sommation d'Euler rentre dans ce cas et l'application de la formule (IV-8) fournit $\alpha_k = \frac{1}{q+1}$,vu que $a_{k,m} = \frac{q^{k-m}}{(q+1)^k} C_k^m$.

La méthode d'Euler admet donc l'algorithme récursif suivant :

$$\begin{cases} t_{k+1}^{(n)} = \frac{1}{q+1} t_k^{(n+1)} + \frac{q}{q+1} t_k^{(n)} \\ t_0^{(n)} = S_n \end{cases}$$

IV-5.2. Algorithme linéaire de losange.

Un certain nombre de procédés de sommation linéaire n'admettent pas de forme récursive deltoïde. Par contre, ils acceptent un algorithme récursif de losange, c'est-à-dire, à 4 termes :

$$\begin{cases} s_{k+1}^{(n)} = \alpha_k s_k^{(n)} + \beta_k s_k^{(n+1)} + \gamma_k s_{k-1}^{(n)} \\ s_{-1}^{(n)} = 0 \qquad s_0^{(n)} = S_n \end{cases}$$

La régularité de l'algorithme est assuré par la condition $\alpha_k + \beta_k + \gamma_k = 1$ et il est possible d'exprimer les éléments a_{ij} de la matrice de sommation en fonction des α_k , β_k et γ_k : il suffit d'introduire la forme $t_k^{(n)} = \sum_{m=0}^{k} a_{km} S_{n+m}$ dans l'algorithme de losange et d'identifier les termes.

On trouve : $\begin{cases} a_{k+1,m} - \alpha_k a_{k,m} - \beta_k a_{k,m-1} - \gamma_k a_{k-1,m} = 0 \\ \alpha_k + \beta_k + \gamma_k = 1 \end{cases}$

IV-6. L'algorithme ε.

Nous abordons, dans ce paragraphe et les suivants, les méthodes d'accélération non linéaires. Elles sont d'invention relativement récentes puisque la doyenne, l'algorithme ε, fut publiée en 1956. Ce qui les rend indispensables n'est pas tant leur puissance accélératrice, car un procédé de sommation linéaire bien choisi peut parfois se révéler très performant, que la plus grande richesse de leur noyau qui les rend aptes à être efficaces dans un plus grand nombre de cas.

IV-6.1. Le procédé Δ^2 d'Aitken.

L'algorithme ε est en fait une généralisation d'un procédé d'accélération connu, depuis 1926, sous le nom de procédé Δ^2 d'Aitken. Ou si on préfère formuler les choses autrement, le procédé Δ^2 d'Aitken apparaît comme la première étape de l'algorithme ε.

Etant donnée une suite S_n, le procédé Δ^2 d'Aitken la transforme en une suite $\varepsilon_2^{(n)}$ qui vaut:

$$\varepsilon_2^{(n)} = \frac{S_n S_{n+2} - S_{n+1}^2}{S_{n+2} - 2S_{n+1} + S_n} \qquad \text{(IV-9)}$$

Une forme déterminante équivalente est très utile pour l'étude théorique du procédé :

$$\varepsilon_2^{(n)} = \frac{\begin{vmatrix} S_n & S_{n+1} \\ \Delta S_n & \Delta S_{n+1} \end{vmatrix}}{\begin{vmatrix} 1 & 1 \\ \Delta S_n & \Delta S_{n+1} \end{vmatrix}}$$

Par exemple, celle-ci permet immédiatement de se rendre compte que le procédé Δ^2 d'Aitken conserve la limite de la suite de départ. Il suffit, pour s'en convaincre, d'y introduire la suite écrite sous la forme: $S_n = S + as(n)$. On trouve, en effet :

$$\varepsilon_2^{(n)} = S + \frac{\begin{vmatrix} as(n) & as(n+1) \\ \Delta as(n) & \Delta as(n+1) \end{vmatrix}}{\begin{vmatrix} 1 & 1 \\ \Delta as(n) & \Delta as(n+1) \end{vmatrix}} = S + as'(n)$$

Par contre, rien dans cette formule ne permet d'affirmer que la suite $\varepsilon_2^{(n)}$ converge plus vite que S_n . Il faudrait, pour cela, que l'on ait : $lim_{n\to\infty} \frac{as'(n)}{as(n)} = 0$, ce qui n'est certainement pas vrai en toute généralité. Le cas as(n) = 1/n fournit un contre-exemple facile à vérifier :

si $S_n \approx S + 1/n + ...$ alors $\varepsilon_2^{(n)} \approx S + 1/(2n) + ...$

On constate que l'asymptote est simplement amortie d'un facteur 2, sans avoir disparu pour autant ; cette amélioration est bien trop minime pour être intéressante en pratique.
La question se pose alors de savoir quel type de suite la transformation $\varepsilon_2^{(n)}$ accélère valablement. C'est la détermination du noyau de la transformation qui permet de répondre à cette question.

Recherchons pour quelle suite S_n on a exactement : $\varepsilon_2^{(n)} = S$, à l'ordre n.

Il faut et il suffit que : $$\begin{vmatrix} S_n & S_{n+1} \\ \Delta S_n & \Delta S_{n+1} \end{vmatrix} = \begin{vmatrix} S & S \\ \Delta S_n & \Delta S_{n+1} \end{vmatrix} \qquad \forall n > N$$

ou encore : $$\begin{vmatrix} S_n - S & S_{n+1} - S \\ S_{n+1} - S_n & S_{n+2} - S_{n+1} \end{vmatrix} = 0$$

ou enfin : $$\begin{vmatrix} S_n - S & S_{n+1} - S \\ S_{n+1} - S & S_{n+2} - S \end{vmatrix} = 0$$

Il faut donc et il suffit qu'il existe deux constantes a_0 et a_1 telles que l'on ait :

$$a_0(S_n - S) + a_1(S_{n+1} - S) = 0 \qquad \forall n > N \qquad \text{(IV-10)}$$

qui est une récurrence linéaire homogène d'ordre un. Sa solution est évidente;
elle s'écrit :

$$S_n = S + \alpha\lambda^n \qquad (\lambda \neq 1)$$

Tel est le noyau du procédé $\Delta^2 \equiv \varepsilon_2^{(n)}$. Il contient toutes les suites exponentiellement convergentes et divergentes quelles que soient les valeurs de α mais aussi de λ. Le progrès est évident par rapport aux méthodes linéaires qui ne pouvaient accélérer que les suites du type $S_n = S + \alpha\lambda^n$ à la condition que le paramètre λ ait une valeur précise et imposée.

Le procédé Δ^2 se présente, en fait, comme une méthode d'extrapolation par une exponentielle: étant donnés trois termes consécutifs d'une suite S_n, S_{n+1} et S_{n+2}, il calcule le meilleur ajustage exponentiel du type:

$$\begin{cases} S_n = S + \alpha\lambda^n \\ S_{n+1} = S + \alpha\lambda^{n+1} \\ S_{n+2} = S + \alpha\lambda^{n+2} \end{cases}$$

On retrouve l'écriture (IV-9) du procédé Δ^2 en résolvant ce système par rapport à S.
Vu la forme du noyau du procédé, on ne s'étonne pas qu'il soit inefficace dans le cas des suites du type $S_n \approx S + \lambda/n + \ldots$.

Remarque: si on applique le procédé d'Aitken à la suite des sommes partielles d'une série de Mac Laurin, $S_n = c_0 + c_1 z + \ldots + c_n z^n$, on trouve :

$$\varepsilon_2^{(n)} = c_0 + c_1 z + \ldots + c_n z^n + \frac{\begin{vmatrix} 0 & c_{n+1} z^{n+1} \\ c_{n+1} z^{n+1} & c_{n+2} z^{n+2} \end{vmatrix}}{\begin{vmatrix} 1 & 1 \\ c_{n+1} z^{n+1} & c_{n+2} z^{n+2} \end{vmatrix}}$$

$$= c_0 + c_1 z + \ldots + c_n z^n + \frac{c_{n+1}^2 z^{n+1}}{c_{n+1} - c_{n+2} z} = [n+1/1]$$

Le procédé Δ^2 fournit donc, dans ce cas, la première ligne de la table de Padé.

IV-6.2. La transformation de Shanks.

Shanks a recherché quelle transformation de suite est susceptible de généraliser la relation (IV-10) sous la forme :

$$\sum_{i=0}^{k} a_i (S_{n+1} - S) = 0 \qquad \forall n > N \tag{IV-11}$$

Il a évidemment trouvé qu'il fallait qu'on ait :

$$\begin{vmatrix} S_n - S & \cdots & S_{n+k} - S \\ S_{n+1} - S & \cdots & S_{n+k+1} - S \\ \vdots & & \vdots \\ S_{n+k} - S & \cdots & S_{n+2k} - S \end{vmatrix} = 0$$

En d'autres termes, il a trouvé une transformation, notée $\varepsilon_{2k}^{(n)}$, qui transforme la suite S_n, obéissant à (IV-11), en la suite constante {S} :

$$\varepsilon_{2k}^{(n)} = \frac{\begin{vmatrix} S_n & S_{n+1} & \cdots & S_{n+k} \\ \Delta S_n & \Delta S_{n+1} & \cdots & \Delta S_{n+k} \\ \vdots & \vdots & & \vdots \\ \Delta S_{n+k-1} & \Delta S_{n+k} & \cdots & \Delta S_{n+2k-1} \end{vmatrix}}{\begin{vmatrix} 1 & 1 & \cdots & 1 \\ \Delta S_n & \Delta S_{n+1} & \cdots & \Delta S_{n+k} \\ \vdots & \vdots & & \vdots \\ \Delta S_{n+k-1} & \Delta S_{n+k} & \cdots & \Delta S_{n+2k-1} \end{vmatrix}} \tag{IV-12}$$

Lorsque k = 1, on retrouve évidemment le procédé Δ^2. Lorsque k > 1, on découvre de nouveaux procédés d'accélération plus puissants que lui. Voyons à quels types de suites ils s'appliquent préférentiellement en déterminant leur noyau.
La relation (IV-11) montre clairement que S_n-S obéit à une récurrence linéaire à coefficients constants dont la solution générale revêt nécessairement la forme d'exponentielles-polynômes. Le noyau de la transformation de Shanks est donc constitué par l'ensemble des suites du type:

$$S_n = S + \sum_{i=1}^{N} p_i(n) e^{z_i n} \qquad \text{(IV-13)}$$

où les $p_i(n)$ sont des polynômes en n et où les z_i sont des nombres quelconques, réels ou complexes. La grosseur relative de ce noyau explique son importance pratique.

Cependant on notera que les suites, fréquemment rencontrées, du type :

$$S_n \approx S + \lambda_1 / n + \lambda_2 / n^2 + \cdots$$

n'en font pas partie, en sorte qu'elles ne bénéficient pas de l'application du procédé de Shanks.

IV-6.3. Transformation de Shanks et approximants de Padé.

Si on applique la transformation de Shanks aux séries de puissances $S_n = \sum_{i=0}^{n} c_i z^i$, en tenant compte de ce que: $\Delta S_n = S_{n+1} - S_n = c_{n+1} z^{n+1}$, on trouve que la relation (IV-12) se réduit à la forme déterminante de l'approximant de Padé [n+k/k]. On a donc l'identité importante:

$$\varepsilon_{2k}^{(n)} = [n + k / k] \qquad \text{(IV-14)}$$

Cette identité a une conséquence importante: la table de Padé n'est utile que dans les cas où la suite $S_n = \sum_{i=0}^{n} c_i z^i$ possède un comportement asymptotique du type (IV-13).

Ceci explique pourquoi la table de Padé est impuissante à accélérer la série dilogarithme $S_n = \sum_{i=0}^{n} \frac{z^i}{(i+1)^2}$ qui se comporte asymptotiquement comme $z^{n+1}/(n+1)$.

C'est l'explication des limitations évoquées au § III-6. On notera que le calcul des quantités $\varepsilon_{2k}^{(n)}$ nécessite la connaissance de 2k+1 termes de la suite, soit : S_n, S_{n+1}, ..., S_{n+2k}.

La transformation de Shanks réalise, en fait, l'extrapolation d'une suite donnée S_n par une somme d'exponentielles-polynômes du type (IV-13) et ajuste automatiquement non seulement les coefficients mais aussi les exposants. Tel est le bénéfice qu'on retire de la non-linéarité de l'algorithme.

IV-6.4. L'algorithme ε.

Dû à Wynn, cet algorithme procède au calcul récursif des quantités $\varepsilon_{2k}^{(n)}$ de la transformation de Shanks définies par la relation (IV-12). Il est rendu indispensable par le fait que les calculs de déterminants sont généralement proscrits, en analyse numérique, à cause des trop fréquentes erreurs de troncature qu'ils provoquent. Rappelons, en effet, que le calcul d'un déterminant $n \times n$ implique en gros n!/2 additions et n!/2 soustractions de quantités voisines.
Les règles de l'algorithme ε sont les suivantes :

$$\begin{cases} \varepsilon_{-1}^{(n)} = 0 \qquad \varepsilon_0^{(n)} = S_n \qquad (n = 0,1,\ldots) \\ \varepsilon_{k+1}^{(n)} = \varepsilon_{k-1}^{(n+1)} + \dfrac{1}{\varepsilon_k^{(n+1)} - \varepsilon_k^{(n)}} \qquad (k,n = 0,1,\ldots) \end{cases} \tag{IV-15}$$

Les quantités d'indice inférieur impair ne sont que des intermédiaires de calcul. Seules les quantités d'indice pair, $\varepsilon_{2k}^{(n)}$, sont intéressantes au point de vue de l'accélération de la convergence des suites. Nous ne reproduisons pas la démonstration de la relation (IV-15) qui exige des manipulations fastidieuses de déterminants.

IV-6.5. Convergence de l'algorithme ε.

Il serait faux de croire que l'algorithme ε ne s'applique avec succès qu'aux suites du type combinaison exponentielles-polynômes (IV-13). Un premier indice, à l'appui de cette thèse, est fourni par la fonction exponentielle pour laquelle on a :

$$S_n = \sum_0^n x^i / i! \approx e^x + O\left(\frac{x^{n+1}}{(n+1)!}\right)$$

On a également remarqué qu'en pratique, les séries alternées étaient bien accélérées par l'algorithme ε : elles correspondent à des suites qui oscillent autour de leur limite. Par contre, les suites qui tendent monotonément vers leur limite se laissent plus difficilement accélérer par l'algorithme ε sauf évidemment si elles sont du type (IV-13).
Cette remarque est illustrée par les deux exemples suivants :

1) $\quad S_n \approx S + \sum_{i=0}^{\infty} \dfrac{a_i}{(n+b)^i} \qquad (a_1 \neq 0)$

L'asymptote principale de cette suite est du type : $\quad as(n) = \dfrac{1}{n+b}$

Pour k fixé, on trouve : $\quad \varepsilon_{2k}^{(n)} \approx S + \dfrac{a_1}{(k+1)(n+k+b)}$

d'asymptote principale : $\quad as'(n) = \dfrac{1}{(k+1)(n+k+b)}$

La suite s'en trouve fort peu accélérée puisque : $\lim_{n\to\infty} \frac{as'(n)}{as(n)} = \frac{1}{k+1} \neq 0$

2) Par contre, dans le cas de la suite oscillante :

$S_n \approx S + (-1)^n \sum_{i=1}^{\infty} \frac{a_i}{(n+b)^i} \qquad (a_i \neq 0)$

d'asymptote principale : $as(n) = \frac{(-1)^n}{n+b}$

on trouve : $\varepsilon_{2k}^{(n)} \approx S + \frac{(-1)^n k!^2 a_1}{2^{2k}(n+b)^{2k+1}}$

avec cette fois : $as'(n) = \frac{(-1)^n k!^2}{2^{2k}(n+b)^{2k+1}}$

et l'accélération est, de toute évidence, spectaculaire.

Cette situation est générale et explique, en particulier, pourquoi les séries de Fourier sont accélérées par l'algorithme ε .

L'étude de la stabilité de l'algorithme ε appelle une remarque similaire : à cause de la différence des quantités $\varepsilon_k^{(n+1)}$ et $\varepsilon_k^{(n)}$ qui intervient dans son écriture (IV-15), il est, en général, instable pour les suites monotones et stable pour les suites oscillantes. Il n'existe toutefois pas de théorème valable en toute généralité dans ce domaine.

IV-6.6 Généralisations de l'algorithme ε.

Deux généralisations de l'algorithme ε existent qui font usage d'une suite auxiliaire x_n .

- Première généralisation : ses règles s'écrivent :

$$\begin{cases} \varepsilon_{-1}^{(n)} = 0 \qquad \varepsilon_0^{(n)} = S_n \qquad (n = 0,1...) \\ \varepsilon_{k+1}^{(n)} = \varepsilon_{k-1}^{(n+1)} + \frac{x_{n+1} - x_n}{\varepsilon_k^{(n+1)} - \varepsilon_k^{(n)}} \qquad (n,k = 0,1,...) \end{cases}$$

Appliquons-les à l'exemple numérique suivant emprunté à Brezinski : soit la suite
$S_n = 1 + 3e^{-1.4x_n}$ avec $x_n = 1.1^{n-1}$ qui converge très lentement vers 1.
On a, en effet, $S_0 = 1.73979...$; $S_1 = 1.64314...$; ... ; $S_4 = 1.38631...$; ...
On trouve $\varepsilon_4^{(0)} = 1.73799...$ par l'algorithme ε simple.
$\varepsilon_4^{(0)} = 1.00272...$ par sa première généralisation.

- Deuxième généralisation : ses règles s'écrivent :

$$\begin{cases} \varepsilon_{-1}^{(n)} = 0 \qquad \varepsilon_0^{(n)} = S_n \qquad (n = 0,1,...) \\ \varepsilon_{k+1}^{(n)} = \varepsilon_{k-1}^{(n+1)} + \dfrac{x_{n+k+1} - x_{n+k}}{\varepsilon_k^{(n+1)} - \varepsilon_k^{(n)}} \qquad (n,k = 0,1,...) \end{cases}$$

Cette variante est également efficace pour l'exemple précédent car on trouve $\varepsilon_4^{(0)} = 1.00224...$

Autre exemple numérique : soit la suite $S_n = n \sin\dfrac{1}{n}$ qui tend vers 1.

On a, par exemple $S_{37} = 0.99987...$
On trouve successivement $\varepsilon_{36}^{(0)} = 0.9999938...$ par l'algorithme ε simple.
$\varepsilon_{36}^{(0)} = 1.0000000027...$ par sa première généralisation et
$\varepsilon_{36}^{(0)} = 0.99999999976...$ par sa deuxième généralisation
à la condition de bien choisir la suite x_n , soit $x_n = \ln(n+1)$.

Ce choix judicieux de la suite auxiliaire x_n correspond évidemment au comportement asymptotique de la suite S_n mais la relation est pratiquement impossible à établir avec certitude dans la mesure où les noyaux des généralisations de l'algorithme ε ne sont pas connus explicitement. En pratique, on se contente de mettre quelques suites caractéristiques au banc d'essai, quitte à changer de méthode si cela ne donne rien.

IV-7. L'algorithme θ.

C'est un des algorithmes les plus puissants de la littérature.
Jusqu'à présent, les algorithmes non linéaires étudiés étaient tous de la forme dite de losange :

$$\begin{cases} \theta_{-1}^{(n)} = 0 \qquad \theta_0^{(n)} = S_n \qquad (n = 0,1,...) \\ \theta_{k+1}^{(n)} = \theta_{k-1}^{(n+1)} + D_k^{(n)} \qquad (n,k = 0,1,...) \end{cases} \qquad \text{(IV-16)}$$

où la forme de $D_k^{(n)}$ variait selon que l'on considérait l'algorithme ρ, ε ou ses généralisations.

C'est par un raisonnement purement intuitif que Brezinski a découvert les règles de l'algorithme θ. Partant du schéma (IV-16) ci-dessus, il a tenté la comparaison des vitesses de convergence des suites $\theta_{2k+2}^{(n)}$ et $\theta_{2k}^{(n)}$ censées converger vers la même limite, la première plus vite que la deuxième. On traduit cette exigence en posant que (Δ porte sur l'indice n) :

$$\lim_{n\to\infty} \frac{\Delta\theta_{2k+2}^{(n)}}{\Delta\theta_{2k}^{(n)}} = 0 \qquad \text{(IV-17)}$$

Cela provient en effet de ce que, lorsqu'une suite T_n converge plus vite que S_n vers la même limite, on a très généralement que :

$$\lim_{n\to\infty} \frac{\Delta T_n}{\Delta S_n} = 0$$

Or les règles (IV-16) donnent :

$$\theta_{2k+2}^{(n)} = \theta_{2k}^{(n+1)} + D_{2k+1}^{(n)}$$

et la condition (IV-17) devient :

$$\lim_{n\to\infty} \frac{\Delta D_{2k+1}^{(n)}}{\theta_{2k}^{(n+1)}} = -1 \qquad \text{(IV-18)}$$

L'astuce consiste maintenant à modifier les règles (IV-16) en introduisant un paramètre w_k , variable avec k que l'on ajuste de façon convenable :

$$\begin{cases} \theta_{-1}^{(n)} = 0 \qquad \theta_0^{(n)} = S_n \\ \theta_{2k+1}^{(n)} = \theta_{2k-1}^{(n+1)} + D_{2k}^{(n)} \\ \theta_{2k+2}^{(n)} = \theta_{2k}^{(n+1)} + w_k D_{2k+1}^{(n)} \end{cases}$$

La condition (IV-18) se transcrit de façon évidente :

$$\lim_{n\to\infty} \frac{\Delta \theta_{2k}^{(n+1)}}{\Delta D_{2k+1}^{(n)}} = -w_k$$

En pratique, cette limite n'est pas connue, d'où l'idée de poser simplement

$$w_k = -\frac{\Delta \theta_{2k}^{(n+1)}}{\Delta D_{2k+1}^{(n)}}$$

en reprenant pour $D_k^{(n)}$, l'expression en vigueur dans l'algorithme ε , soit :

$$D_k^{(n)} = \frac{1}{\theta_k^{(n+1)} - \theta_k^{(n)}}$$

Tenant compte de toutes ces indications, on trouve les règles définitives de l'algorithme θ , où l'opérateur Δ agit sur la variable n :

$$\theta_{-1}^{(n)} = 0 \qquad \theta_0^{(n)} = S_n \qquad (n = 0,1,\cdots)$$

$$\theta_{2k+1}^{(n)} = \theta_{2k-1}^{(n+1)} + \frac{1}{\Delta\theta_{2k}^{(n)}}$$

$$\theta_{2k+2}^{(n)} = \frac{\Delta[\theta_{2k}^{(n+1)}\Delta\theta_{2k+1}^{(n)}]}{\Delta^2\theta_{2k+1}^{(n)}}$$

Les θ d'indice inférieur impair sont des intermédiaires de calcul et seuls les θ d'indice pair sont retenus.

Il importe de noter que la démarche suivie est purement intuitive et que la seule justification véritable de cet algorithme réside, a posteriori, dans son extrême efficacité. Facile à programmer, cet algorithme est très difficile à étudier du point de vue théorique. On note que :

- Ce n'est plus un algorithme de losange. Par exemple, le calcul de $\theta_2^{(n)}$, première étape de l'algorithme, nécessite la connaissance de quatre termes consécutifs : S_n , S_{n+1} , S_{n+2} et S_{n+3} alors que les algorithmes ε ou ρ n'avaient besoin que de trois termes successifs.
- Sauf si k = 1, il n'existe pas de forme déterminante pour $\theta_{2k}^{(n)}$ comme c'était le cas pour $\varepsilon_{2k}^{(n)}$, (c'était la transformation de Shanks) ou pour $\rho_{2k}^{(n)}$ (nous ne l'avons pas donnée mais elle existe). C'est cette carence qui rend difficile l'étude théorique de l'algorithme θ sauf dans sa première étape.
- Pour la même raison, il semble impossible de déterminer le noyau de l'algorithme θ , sauf dans sa première étape. On a en effet que :

$$\theta_2^{(n)} = \frac{\begin{vmatrix} S_n & S_{n+1} & S_{n+2} \\ \Delta S_n & \Delta S_{n+1} & \Delta S_{n+2} \\ 2\Delta S_n & \Delta S_{n+1} & 0 \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 1 \\ \Delta S_n & \Delta S_{n+1} & \Delta S_{n+2} \\ 2\Delta S_n & \Delta S_{n+1} & 0 \end{vmatrix}} = \frac{\Delta\left[S_{n+1}\Delta\frac{1}{\Delta S_n}\right]}{\Delta^2\left[\frac{1}{\Delta S_n}\right]}$$

$\theta_2^{(n)} = S \qquad \forall n$ ssi :

$$\Delta\left[(S_{n+1} - S)\Delta\frac{1}{\Delta S_n}\right] = 0 \qquad d'où \qquad (S_{n+1} - S)\Delta\frac{1}{\Delta S_n} = \nu$$

où ν est une constante arbitraire.

$$(S_{n+1} - S)\left[\frac{1}{S_{n+2} - S_{n+1}} - \frac{1}{S_{n+1} - S_n}\right] = \nu$$

que l'on réécrit, un peu différemment, sous la forme :

$$\frac{S_{n+1}-S}{S_{n+2}-S_{n+1}}-\left[\frac{S_{n+1}-S}{S_{n+1}-S_n}-1\right]=\nu+1$$

Si on pose $\tau_n=\frac{S_n-S}{S_{n+1}-S_n}$

l'équation précédente devient : $\tau_{n+1}-\tau_n=\nu+1$, récurrence inhomogène du premier ordre à coefficients constants dont la solution générale s'écrit :

$$\tau_n=(\nu+1)n+\mu$$

On trouve donc :

$$1+\frac{1}{\tau_n}=\frac{S_{n+1}-S}{S_n-S}=\frac{(\nu+1)n+(\mu+1)}{(\nu+1)n+\mu} \qquad \text{(IV-19)}$$

Cette récurrence d'ordre 1 qui porte sur S_n - S, se résout à nouveau sans difficulté. En résumé, on trouve que les suites S_n qui constituent le noyau de la première étape de l'algorithme θ s'écrivent sous l'une des formes :

$$S_n=S+\lambda\frac{\Gamma(n+a)}{\Gamma(n+b)} \qquad (si \quad \nu\neq -1)$$

$$S_n=S+\lambda\rho^n \qquad (si \quad \nu=-1)$$

On comparera ce noyau à celui de la première étape de l'algorithme ε qui s'écrivait simplement $S_n=S+\lambda\rho^n$ et on mesure le progrès accompli par l'algorithme θ. En particulier, on comprend pourquoi les suites du type $S_n\approx S+\frac{\lambda}{n+b}+\cdots$ sont bien accélérées par l'algorithme θ puisqu'elles appartiennent au noyau avec le choix b = a + 1.

IV-8. L'algorithme E.

Dû à Havie et Brezinski, l'algorithme E permet de comprendre comment fonctionnent les méthodes non-linéaires d'accélération de la convergence.
Supposons que l'on connaisse l'échelle asymptotique complète d'une suite donnée S_n :

$$S_n=S+a_1g_1(n)+a_2g_2(n)+a_3g_3(n)+\cdots$$

avec : $\lim_{n\to\infty}\frac{g_{i+1}(n)}{g_i(n)}=0 \qquad \forall i$

L'idée vient tout naturellement à l'esprit d'extrapoler la suite S_n au moyen de son échelle asymptotique propre. Il y a lieu, pour cela, de résoudre le système :

$$S_{n+i} = S + a_1 g_1(n+i) + \cdots + a_k g_k(n+i) \qquad i = 0,1,...,k$$

C'est un système de k + 1 équations pour les k + 1 inconnues : a_1, ..., a_k et S. Sauf le cas exceptionnel où S_n serait exactement égale à : $S + a_1 g_1(n) + \cdots + a_k g_k(n)$, la solution obtenue en résolvant ce système dépend de n **et de k**. Notons $E_k^{(n)}$, la valeur obtenue pour S de cette façon. La règle de Cramer permet de trouver une forme déterminante pour $E_k^{(n)}$:

$$E_k^{(n)} = \frac{\begin{vmatrix} S_n & \cdots & S_{n+k} \\ g_1(n) & \cdots & g_1(n+k) \\ \vdots & & \vdots \\ g_k(n) & \cdots & g_k(n+k) \end{vmatrix}}{\begin{vmatrix} 1 & \cdots & 1 \\ g_1(n) & \cdots & g_1(n+k) \\ \vdots & & \vdots \\ g_k(n) & \cdots & g_k(n+k) \end{vmatrix}} = \frac{dtm(S_n \quad g_1(n) ... g_k(n))}{dtm(1 \quad g_1(n) ... g_k(n))} \qquad \text{(IV-20)}$$

On voit, grâce à cette forme déterminante, que , si les fonctions $g_i(n)$ font réellement partie de l'échelle asymptotique propre de S_n, la suite $E_k^{(n)}$ convergera vers S beaucoup plus vite que S_n. En effet, vu les propriétés des déterminants, on trouve que le comportement asymptotique de la suite $E_k^{(n)}$ est de l'ordre de grandeur de la première asymptote négligée,

soit : $E_k^{(n)} \approx S + 0(g_{k+1}(n))$

Ceci représente l'optimum imaginable, quelle que soit la procédure accélératrice employée. On voit donc que, lorsqu'on doit accélérer une suite S_n, l'idéal est d'en connaître l'échelle asymptotique propre. La connaissance des fonctions $g_i(n)$ suffit : il n'est pas nécessaire de connaître les coefficients a_i qui pondèrent les asymptotes. Cela est heureux car ceux-ci sont d'accès beaucoup plus difficile en pratique. Si certains coefficients a_i étaient connus, il est clair qu'on pourrait regrouper plusieurs fonctions $g_i(n)$ en une seule asymptote et l'accélération s'en trouverait accrue mais ce cas est très rare.

Lorsque l'échelle asymptotique est inaccessible ou simplement inconnue, il convient d'envisager une des stratégies suivantes :

- soit essayer une échelle asymptotique qu'on se donne, à priori, en espérant qu'elle ne sera pas trop éloignée de l'échelle véritable. Il y a lieu de noter que, quand le choix effectué ne donne pas satisfaction, on peut toujours essayer une des douze généralisations mises au point par Brezinski qui sont à l'algorithme E ce que θ est à ε . Ces généralisations ont pour but de tenter de rectifier

le choix médiocre effectué pour les g_i (n). Nous entrons ici dans un domaine trop spécialisé pour qu'il figure en détail dans ce cours.

- soit recourir à une forme implicite de l'algorithme. Par exemple, l'analogie entre l'équation (IV-20) et la transformation de Shanks indique qu'il suffit de poser : $g_i(n) = \Delta s_{n+i-1}$ dans (IV-20) pour retrouver l'algorithme ε qui apparaît, de ce fait, comme un cas particulier de l'algorithme E, où on considère que la suite S_n se comporte asymptotiquement comme :

$$S_n \approx S + a_1 \Delta S_n + a_2 \Delta S_{n+1} + \ldots + a_k \Delta S_{n+k-1}$$

La forme est dite implicite parce que S_n figure dans l'écriture des asymptotes $g_i(n)$.
On retrouve bien la récurrence à coefficients constants qui définit le noyau de l'algorithme ε et dont la solution générale est une somme d'exponentielles-polynômes.

La généralité de l'algorithme est telle qu'il englobe beaucoup de procédés connus d'accélération. Un autre exemple est la transformation de Richardson qui correspond au choix $g_i(n) = x_n^{\,i}$. Nous verrons plus loin d'autres exemples encore.

En résumé, l'algorithme E propose un choix : soit recourir à une échelle asymptotique explicite, qu'elle soit exacte (cas optimal) ou approchée, ou choisir une échelle asymptotique implicite où les S_i interviennent dans l'écriture des $g_i(n)$. Ce dernier cas est pratiqué toutes les fois qu'on n'a aucune idée du comportement asymptotique véritable de la suite S_n . On espère ainsi que le choix implicite se rapproche de la réalité. La grande variété des algorithmes existants n'a d'autre justification que d'offrir le plus vaste éventail de méthodes plus ou moins aptes à accélérer des suites aux comportements asymptotiques les plus divers. Cela signifie également qu'il n'existe pas d'algorithme d'accélération universelle puisque toute procédure doit être adaptée au comportement asymptotique de la suite considérée.

IV-9.1. L'algorithme E récursif.

Il n'est jamais recommandé de calculer de grands déterminants du type (IV-20). Heureusement, il existe un algorithme récursif qui calcule les quantités $E_k^{(n)}$. Il se note :

$$\begin{cases} E_0^{(n)} = S_n \qquad g_{0,i}^{(n)} = g_i(n) \qquad \begin{pmatrix} n = 0,1,\ldots \\ i = 1,2,\ldots \end{pmatrix} \\ E_k^{(n)} = \dfrac{E_{k-1}^{(n)} g_{k-1,k}^{(n+1)} - E_{k-1}^{(n+1)} g_{k-1,k}^{(n)}}{g_{k-1,k}^{(n+1)} - g_{k-1,k}^{(n)}} \\ g_{k,i}^{(n)} = \dfrac{g_{k-1,i}^{(n)} g_{k-1,k}^{(n+1)} - g_{k-1,i}^{(n+1)} g_{k-1,k}^{(n)}}{g_{k-1,k}^{(n+1)} - g_{k-1,k}^{(n)}} \qquad (k \geq 1, n \geq 0, i \geq k+1) \end{cases} \tag{IV-21}$$

IV-9.2. Illustration d'une échelle asymptotique exacte.

Considérons l'exemple de la suite : $S_n = (PIC)_n$:

$$S_n = \sum_{i=0}^{n} \frac{1}{(i+1)^2} = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \cdots + \frac{1}{(n+1)^2}$$

Cette suite tend vers $\pi^2/6 = 1.644934...$
Nous avons vu au paragraphe IV-1. que son comportement asymptotiques est donné par :

$S_n \approx S + \lambda_1 / n + \lambda_2 / n^2 + \cdots$ d'où $g_i(n) = n^{-i}$.

L'algorithme optimal prescrit donc de calculer les expressions suivantes :

$$E_k^{(n)} = \frac{dtm\left(S_n, \frac{1}{n+\gamma}, \frac{1}{(n+\gamma)^2}, ..., \frac{1}{(n+\gamma)^k}\right)}{dtm\left(1, \frac{1}{n+\gamma}, \frac{1}{(n+\gamma)^2}, ..., \frac{1}{(n+\gamma)^k}\right)}$$

Le dénominateur se calcule sans peine par la règle de Van der Monde. Il reste à développer le numérateur selon les éléments de la première colonne et à utiliser la même règle pour obtenir le résultat du quotient final :

$$E_k^{(n)} = \sum_{m=0}^{k} \frac{(-1)^{m+k}}{k!} (n+m+\gamma)^k C_k^m S_{n+m}$$

On retrouve le procédé de Salzer qui apparaît comme optimal pour les suites qui se comportent comme :

$S_n \approx S + a_1 / n + a_2 / n^2 + \cdots$

On peut calculer les $E_k^{(n)}$ et les disposer dans une table de ce type :

$E_0^{(0)} = S_0$	$E_1^{(0)}$	$E_2^{(0)}$	$E_3^{(0)}$	$E_4^{(0)}$	...
$E_0^{(1)} = S_1$	$E_1^{(1)}$	$E_2^{(1)}$	$E_3^{(1)}$	...	...
$E_0^{(2)} = S_2$	$E_1^{(2)}$	$E_2^{(2)}$	...	...	...
$E_0^{(3)} = S_3$	$E_1^{(3)}$	...	...	...	...
$E_0^{(4)} = S_4$	...	...	...	...	...
...	...	...	...	...	...

Table IV-1.

Dans le cas de l'exemple numérique étudié, on trouve les valeurs, $E_k^{(n)}$, suivantes ainsi que le tableau des précisions :

1	3/2	13/8	355/216	1895/1152
5/4	19/12	59/36	1421/864	
49/36	29/18	473/288		
205/144	1169/720			
5269/3600				

Table IV-2.

0.9	2.4	4.4	7.1	10.9
1.4	3.3	5.6	8.8	
1.7	3.9	6.5		
2.0	4.3			
2.2				

Table IV-3.

L'accélération est optimale pour le terme $E_4^{(0)}$ qui s'avère correct avec cinq chiffres décimaux. On voit donc que l'information que constituent ces cinq chiffres est déjà présente dans les cinq premiers termes de la suite S_0, S_1, ..., S_4 .

IV-8.3. Application de l'algorithme E au calcul des intégrales définies par la méthode des trapèzes.

La méthode des trapèzes permet de calculer des intégrales définies grâce à la formule suivante :

$$I = \int_0^1 f(x)dx = \lim_{n\to\infty} S_n \qquad \text{où} \qquad S_n = \frac{1}{n}\sum_{k=0}^{n}{}'' \, f(\frac{k}{n})$$

Σ" signifie que l'on compte les termes k = 0 et k = n pour moitié seulement. On montre, en analyse numérique, que la suite S_n se comporte asymptotiquement comme :

$$S_n \approx S + \frac{C_1}{n^2} + \frac{C_2}{n^4} + \frac{C_3}{n^6} + ...$$

avec :

$$C_1 = \frac{1}{12}(f'(1) - f'(0))$$

$$C_2 = -\frac{1}{720}(f'''(1) - f'''(0)) \qquad etc...$$

En application de l'algorithme E, on trouve la formule sommatoire suivante :

$$E_k^{(n)} = \frac{dtm\left(S_n \quad n^{-2}n^{-4}...n^{-2k}\right)}{dtm\left(1 \quad n^{-2}n^{-4}...n^{-2k}\right)}$$

Les déterminants se calculent à nouveau par la règle de Van der Monde et on trouve :

$$E_k^{(n)} = \frac{\sum_{j=0}^{k}(-1)^j S_{n+j} C_k^j (n+j)^{2k+1} \frac{(2n+j-1)!}{(2n+k+j)!}}{\sum_{j=0}^{k}(-1)^j C_k^j (n+j)^{2k+1} \frac{(2n+j-1)!}{(2n+k+j)!}} = 2\sum_{j=0}^{k}(-1)^{k+j} \frac{(2n+j-1)!(n+j)^{2k+1}}{j!(k-j)!(2n+k+j)!} S_{n+j}$$

Exemple : Soit à calculer l'intégrale : $\int_0^1 \frac{dx}{x+1} = ln\,2 = 0.6931471805599453094...$

Douze termes dans la suite fournissent treize chiffres décimaux corrects :

$E_{11}^{(0)} =$ **0.693147180559888**...

IV-8.4. Les transformations de Levin.

On pourrait reproduire le schéma précédent sur un grand nombre d'échelles asymptotiques afin de déduire, chaque fois, de (IV-20), la formule sommatoire qui s'applique. Il suffit que les déterminants soient calculables exactement pour que le calcul récursif devienne superflu. Par exemple, la méthode d'interpolation polynomiale de Richardson se retrouve par ce procédé, en posant, dans (IV-20), $g_i(n) = x_n^1$ et en calculant à nouveau les déterminants par la formule de Van der Monde.

Un autre exemple, d'une importance extrême, est constitué par les transformations de Levin qui, avec l'algorithme θ, s'avèrent être actuellement les meilleurs algorithmes automatiques d'accélération de la convergence. Pour obtenir les deux transformations de Levin nommées u et t, on commence par faire le choix d'une forme générale pour les $g_i(n)$: $g_i(n) = x_n^{i-1}\xi_n$

où la suite x_n est à éléments distinct et ξ_n est à éléments quelconques.

Ce choix permet de calculer les déterminants qui définissent $E_k(n)$ grâce à la règle des mineurs et celle de Van der Monde :

$$E_k^{(n)} = \frac{\sum_{m=0}^{k} \frac{S_{n+m}}{\xi_{n+m}} \pi_n^{k,m}}{\sum_{m=0}^{k} \frac{1}{\xi_{n+m}} \pi_n^{k,m}} \qquad \text{(IV-22)}$$

où les $\pi_n^{k,m} = \sum_{\substack{r=0 \\ r\neq m}}^{k} \frac{1}{x_{n+r} - x_{n+m}}$

Reste à trouver un bon choix pour les suites auxiliaires x_n et ξ_n .
Levin en a trouvé deux qui constituent les transformations, t et u, qui portent son nom.

a) La transformation t de Levin.
Elle se déduit de (IV-22) en posant : $\xi_n = \Delta S_n$ et $x_n = 1/(n+1)$
On trouve la suite transformée t (pour rappel : $t_0^{(n)} = S_n$) :

$$t_k^{(n)} = \frac{\sum_{m=0}^{k} \frac{S_{n+m}}{\Delta S_{n+m}} (-1)^m C_k^m (n+m+1)^{k-1}}{\sum_{m=0}^{k} \frac{1}{\Delta S_{n+m}} (-1)^m C_k^m (n+m+1)^{k-1}}$$

b) La transformation u de Levin.
Elle se déduit de (IV-22) en posant : $\xi_n = (n+1)\,\Delta S_n$ et $x_n = 1/(n+1)$
On trouve la suite transformée u (pour rappel : $u_0^{(n)} = S_n$) :

$$u_k^{(n)} = \frac{\sum_{m=0}^{k} \frac{S_{n+m}}{\Delta S_{n+m}} (-1)^m C_k^m (n+m+1)^{k-2}}{\sum_{m=0}^{k} \frac{1}{\Delta S_{n+m}} (-1)^m C_k^m (n+m+1)^{k-2}}$$

La transformation t agit mieux sur les suites oscillantes alors que u est efficace sur les suites monotones. Il ne faut guère chercher de justification théorique rigoureuse au choix des suites auxiliaires qui ont engendré ces algorithmes dont on n'a même pas pu démontrer qu'ils étaient réguliers. L'expérience numérique révèle cependant qu'ils sont efficaces et c'est ce qui leur confère toute leur importance.

Le noyau de la transformation, u, par exemple, se détermine assez aisément en posant $u_k^{(n)} = S \quad \forall n$ d'où

$$\sum_{m=0}^{k} \frac{S_{n+m} - S}{\Delta S_{n+m}} (-1)^m C_k^m (n+m+1)^{k-2} = 0$$

Posons encore : $h_n = \frac{S_n - S}{\Delta S_n} (n+1)^{k-2}$ (IV-23)

Il vient : $\sum_{m=0}^{k} h_{n+m} (-1)^m C_k^m = 0$

récurrence d'ordre k, à coefficients constants, d'équation caractéristique $(z-1)^k = 0$.

Il en résulte que : $h_n = a_0 + a_1 n + ... + a_{k-1} n^{k-1}$ et on a la possibilité de remonter à S_n en sommant la récurrence (IV-23) qui est d'ordre 1. Par exemple, le noyau de la deuxième étape (k = 2) apparaît identique à celui de θ_2 . On a, en effet, successivement : $h_n = a_0 + a_1 n$, d'où :

$$\frac{S_n - S}{S_{n+1} - S_n} = a_0 + a_1 n \qquad \Rightarrow 1 + \frac{1}{a_0 + a_1 n} = \frac{S_{n+1} - S}{S_n - S} = \frac{a_1 n + (a_0 + 1)}{a_1 n + a_0} = \frac{n + \lambda}{n + \mu} \qquad si \quad a_1 \neq 0$$

$$= \rho \qquad si \quad a_1 = 0$$

si $a_1 = 0$, on trouve un morceau de noyau : $S_n = S + A\rho^n$

Si $a_1 \neq 0$, on trouve l'autre morceau : $S_n = S + A\frac{\Gamma(n+\lambda)}{\Gamma(n+\mu)}$

IV-9. Algorithmes pour suites itératives.

Il existe une catégorie de suites qui apparaissent fréquemment en pratique, ce sont les suites itératives. Considérons l'exemple suivant : soit à trouver une approximation de la racine proche de 1.4 de l'équation cubique : $x^3 + 2x^2 + 10x - 20 = 0$. Sa vraie valeur est : x = 1.368808107... Plusieurs schémas itératifs sont envisageables. Nous en avons retenu deux :

$$S_{n+1} = \frac{20}{S_n^2 + 2S_n + 10} \qquad S_0 = 1 \quad (par \ \ exemple)$$

$$T_{n+1} = \frac{20 - 2T_n^2 - T_n^3}{10} \qquad T_0 = 1 \quad (idem)$$

La suite S_n converge lentement : S_n = {1, 1.358, 1.295, 1.401, 1.354, ...} .
La suite T_n diverge.
Peut-on espérer trouver une méthode accélératrice pour S_n et pourquoi pas une méthode sommatoire pour T_n ? Deux algorithmes existent à ces fins.

IV-9.1. La transformation GBW.

Due à Germain-Bronne et Wimp, elle est basée sur le recours à un algorithme de type deltoïde :

$$\begin{cases} \sigma_{k+1}^{(n)} = \dfrac{x_n \sigma_k^{(n+1)} - x_{n+k+1} \sigma_k^{(n)}}{x_n - x_{n+k+1}} \qquad (n,k \geq 0) \\ \sigma_0^{(n)} = S_n \end{cases} \qquad \text{(IV-24)}$$

où la suite x_n est à particulariser convenablement.
Rappelons que ce schéma n'est rien d'autre que la forme récursive de l'algorithme de triangle :

$$\sigma_k^{(n)} = \sum_{m=0}^{k} \mu_{km} S_{n+m}$$

avec $\mu_{km} = \prod_{i=0, i\neq m}^{k} \frac{x_{n+i}}{x_{n+i} - x_{n+m}}$ $\qquad (et \ \sum \mu_{km} = 1)$

Précisément, le choix qui convient pour x_n est : $x_n = \Delta S_n$. Il permet de construire un algorithme dont le noyau est très différent de ceux étudiés jusqu'à présent. La sommation par (IV-24) , avec $x_n = \Delta S_n$ est exacte si S_n est de la forme

$$S_n = S + \lambda_1 \Delta S_n + \lambda_2 (\Delta S_n)^2 + \cdots + \lambda_k (\Delta S_n)^k \qquad \text{(IV-25)}$$

En d'autres termes, si S_n est solution de cette récurrence, on peut montrer que $\sigma_k^{(n)} = S \ \forall$ n . Malheureusement, on ne peut pas expliciter les solutions de cette récurrence qui est hautement non-linéaire. On note qu'un algorithme équivalent s'obtient en appliquant l'algorithme E avec le choix suivant des g_i (n) : $g_i(n) = (\Delta S_n)^i$

L'intérêt de la définition (IV-25) du noyau de la transformation repose sur l'argumentation suivante : supposons que nous soyons intéressés par l'accélération d'une suite itérative définie par :

$S_{n+1} = \phi(S_n)$.

Rappelons que cette récurrence tend vers une racine de $z = \phi(z)$ aux conditions que :

- que $|\phi'(z)| < 1$, au voisinage de S
- que S_0 soit suffisamment proche de S

Si on développe ϕ en série de Taylor, au voisinage de S, on obtient :

$$S_{n+1} = S + c_1 (S_n - S) + c_2 (S_n - S)^2 + \cdots$$

posant : $R_n = S_n - S$, on trouve : $\Delta R_n = (c_1 - 1) R_n + c_2 R_n^2 + \cdots$

Inversant ce développement, on trouve : $R_n = c_1' \Delta R_n + c_2' (\Delta R_n)^2 + c_3' (\Delta R_n)^3 + \cdots$

c'est-à-dire : $S_n = S + c_1' \Delta S_n + c_2' (\Delta S_n)^2 + \cdots$

On retrouve la définition (IV-25) du noyau de la transformation GBW. On s'attend donc à ce que celui-ci fonctionne sur les suites itératives. Les applications numériques le confirment au-delà de tout espoir et on a, en plus, que l'algorithme fonctionne également comme méthode sommatoire lorsque la série itérative ne diverge pas trop violemment.

IV-9.2. Le procédé d'Overholt.

La littérature mentionne également cette autre méthode apparentée à GBW mais qui lui semble inférieure en pratique. Elle est basée sur un algorithme deltoïde du type (IV-24) avec un autre choix de la suite x_n . Plus précisément, ses règles s'énoncent :

$$\begin{cases} V_{k+1}^{(n)} = \dfrac{(\Delta S_{n+k-1})^{k+1} V_k^{(n+1)} - (\Delta S_{n+k})^{k+1} V_k^{(n)}}{(\Delta S_{n+k-1})^{k+1} - (\Delta S_{n+k})^{k+1}} \\ V_0^{(n)} = S_n \end{cases}$$

Les résultats théoriques manquent sur cet algorithme.

IV-9.3. Méthode de Steffensen.

Pour clore ces paragraphes consacrés aux suites itératives, signalons une approche très différente et encore plus puissante.
Devant calculer une racine de l'équation : $x = \phi(x)$, on peut soit :

- recourir au schéma itératif classique $x_{n+1} = \phi(x_n)$, quitte à utiliser conjointement l'algorithme GBW pour en accélérer la convergence.

- recourir à des algorithmes du type Newton ou Laguerre qui requièrent le calcul d'une ou plusieurs dérivées de $\phi(x)$.

- recourir à des méthodes plus élaborées qui consistent à combiner le schéma itératif $x_{n+1} = \phi(x_n)$ avec un algorithme d'accélération classique. Par exemple, si on le combine, avec la première étape de l'algorithme ε, on a besoin de trois termes consécutifs de la suite définie par:

$x_{n+1} = \phi(x_n)$, c'est-à-dire :

S_n, $S_{n+1} = \phi(S_n)$, et $S_{n+2} = \phi(S_{n+1}) = \phi(\phi(S_n))$

On peut alors imaginer remplacer l'itération $x_{n+1} = \phi(x_n)$ par cette autre, nettement plus performante, dite de Steffensen :

$$S_{n+1} = \frac{S_n \phi(\phi(S_n)) - \phi^2(S_n)}{\phi(\phi(S_n)) - 2\phi(S_n) + S_n} \qquad \text{(IV-26)}$$

Exemple numérique : Considérons l'itération suivante :

$$\begin{cases} S_{n+1} = S_n^2 - 1 \\ S_0 = 1.5 \end{cases}$$

destinée à fournir la racine de l'équation $x^2 - x - 1 = 0$, qui vaut $(1 + \sqrt{5})/2 = 1.618033989...$

En fait, cette suite S_n diverge. Par contre, l'algorithme de Steffensen converge et il le fait même plus vite que ne le ferait l'algorithme GBW en resommant la suite divergente.
Naturellement, le croisement que nous avons fait entre l'itération classique $x_{n+1} = \phi(x_n)$ et le procédé Δ^2 peut être généralisé à d'autres algorithmes plus puissants, par exemple, à l'algorithme ε , qui généralise le procédé Δ^2 .

IV-11. Performances comparées des divers algorithmes étudiés.

Nous avons testé les divers algorithmes étudiés sur les suites prédéfinies au paragraphe IV-1.
Dans chaque cas, nous avons retenu treize termes dans la suite à accélérer $\{S_0, S_1, \ldots, S_{12}\}$. L'optimum atteint par chaque algorithme est indiqué dans le tableau IV-4 ci-dessous, par le nombre p de chiffres népériens exacts obtenus.

Suites	S_{12}	ε	θ	C/1	C/2	Eul	Salz	ρ_{n+1}	L/t	L/u	gbw	Ov
$\sum_0^n (-1)^k/(k+1)$	2.9	22.4	29.4	6.1	4.3	15.3	D	D	32.4	31.9	9.7	10.2
$\sum_0^n (-1)^k/\sqrt{k+1}$	1.4	22.5	27.9	5.4	3.2	15.8	D	D	32.6	32.7	16.3	11.4
$\sum_0^n 1/(k+1)^2$	3.1	4.4	25.5	D	D	D	28	29.9	4.8	26.6	4.3	4.5
$\sum_0^n (-1)^k k!$	D	7.1	15.5	D*	D*	D*	D	D*	16.7	15.9	1.8	D
$\sum_0^n \sin(k)/k$	3	7.5	3.8	5.1	4.3	4.5	D	D	3.5	3.5	D	4
$S_{n+1} = \frac{20}{S_n^2 + 2S_n + 10}$	11.0	48.6	34.3	D	D	15.5	D	D	22.6	21.8	74	73.2
$S_{n+1} = \frac{20 - 2S_n^2 - S_n^3}{10}$	DO	2.8	3.0	2.7	3.6	2.9	D	0.9	2.7	2.7	16.0	4.1
$\sum_0^n (-1)^{[k/2]}/(k+1)$	6.0	11.8	D	D	D	8.6	D	DO	D	D	D	D

Table IV-4. Légende : D(O) : divergence (oscillante)
D* : divergence atténuée.

On peut faire les commentaires suivants :

- comme prévu, les procédés linéaires, Césaro C/1 et C/2, ou Euler, ne sont jamais indispensables quoique ce dernier sauve l'honneur avec la suite $(LUB)_n$.

- θ et Levin/u sont en concurrence étroite avec un léger avantage à ce dernier qui le fait préférer. De même, on peut généralement écarter Levin/t au profit de Levin/u.

- GBW et Overholt ont des performances fort semblables sur les suites itératives, avec un avantage au premier que l'on retiendra donc.

- Les séries de Fourier ne se laissent accélérer, encore que modérément, que par l'algorithme ε . On étudiera, au chapitre V, une alternative basée sur l'algorithme E.