EISTI
CERGY - PAU

Correction des exercices du TD 2

Exercice 2.1 et 2.3

1. Calculer pour le standard IEEE-754 en simple précision
   (a) le biais,caractérisant l'intervalle de variation de la valeur de l'exposant $B = 2^{q-1} - 1$ où q représente le nombre de bits de l'exposant ;
   (b) les valeurs l_- et l_+, respectivement égales à $l_- = B + 1$ et $l_+ = B$;
   (c) la valeur de la plus grande mantisse ;
   (d) le plus grand nombre que nous pouvons représenter ;
   (e) le plus petit nombre positif que nous pouvons représenter ;
   (f) la précision des valeurs numériques représentées par ce standard
2. Répéter pour la double précision les calculs précédents.

Comparaison de différentes quantités dans le standard IEEE-754 suivant la simple ou double précision.
On rappelle les données :

donnée et nombre de bits	simple précision	double précision
s	1	1
p	23	52
q	8	11

question	quantité calculée	Simple précision	Double précision
1	Biais $B = 2^{q-1} - 1$	$2^{8-1} - 1 = 127$	$2^{11-1} - 1 = 1023$
2	$l_- = -biais + 1$	$-127 + 1 = -126$	$-1023 + 1 = 1022$
	$l_+ = biais$	127	1023
3	plus grande mantisse	$\underbrace{111...1}_{23\text{ fois}}$	$\underbrace{111...1}_{52\text{ fois}}$
4	plus grand nombre représentable	$\underbrace{11111110}_{8\text{ bits}}\ \underbrace{111...1}_{23\text{ fois}}$	$\underbrace{11111111110}_{11\text{ bits}}\ \underbrace{111...1}_{52\text{ fois}}$
5	plus petit nombre positif représentable	$\underbrace{00000001}_{8\text{ bits}}\ \underbrace{000...0}_{23\text{ fois}}$	$\underbrace{00000001}_{11\text{ bits}}\ \underbrace{000...0}_{52\text{ fois}}$
6	précision = epsilon machine $\varepsilon = 2^{-p}$	$\varepsilon = 2^{-p} = 2^{-23} = 1,192.10^{-7}$	$\varepsilon = 2^{-p} = 2^{-52} = 2,22.10^{-16}$

Le plus grand nombre représentable peut se calculer. En simple précision, il correspond à $1,\underbrace{111...1}_{23\text{ fois}} \times 10^{11111110}_{(2)} = \left(1 + \sum_{i=1}^{23} 2^{-i}\right).2^{2^8-2-127} = 1,9999998808.2^{127} = 3,4.10^{38}$. En double précision, on obtient $1,\underbrace{111...1}_{52\text{ fois}} \times 10^{11111111110} = \left(1 + \sum_{i=1}^{52} 2^{-i}\right).2^{2^{11}-2-1023} = 1,7976931348.10^{308}$.

Le plus petit nombre positif représentable peut se calculer. En simple précision, il correspond à $1,\underbrace{000...0}_{23\text{ fois}} \times 10^{00000001}_{(2)} = 2^{1-127} = 2^{-126} = 1,17549435.10^{-38}$. En double précision, on obtient $1,\underbrace{000...0}_{52\text{ fois}} \times 10^{00000001}_{(2)} = 2^{1-1023} = 2^{-1022} = 2,2250738.10^{-308}$

Exercice 2.2 et 2.4

Représentation au standard IEEE-754 de $5,75$ et $0,1$, en simple et double précision.

1. Cas de $5,75$

   Partie entière : $5 = 4 + 1 = 2^2 + 2^0 = 101_{(2)}$.
   Partie décimale (en reprenant les résultats de l'ex. 1.5). $0,75_{(10)} = 2^{-1} + 2^{-2} = 0,11_{(2)}$
   Nombre : $5,75 = 101,11_{(2)} = 1,0111.10^{10}_{(2)} = 1,0111_{(2)}.2^2_{(10)}$ représenté en simple précision, en décalant l'exposant de 127 par $2^{2+127}_{(10)} = 2^{129}_{(10)} = 10000001_{(2)}$
   Le même nombre est représenté en double précision, en décalant l'exposant de 1023 donc avec $2^{2+1023}_{(10)} = 2^{1025}_{(10)} = 10000000001_{(2)}$
   On en déduit :

   - en simple précision : $\underbrace{10000001}_{8\text{ bits}}\ \underbrace{01110...0}_{23\text{ bits}}$
   - en double précision $\underbrace{10000000001}_{11\text{ bits}}\ \underbrace{01110...0}_{52\text{ bits}}$

   Il n'y a aucune erreur de représentation, ni en simple ni en double précision.

2. Cas de $0,1$

   Partie entière : nulle
   Partie décimale (en reprenant les résultats de l'ex. 1.5). $0,1_{(10)} = 0,000\underline{1100}_{(2)} = 1,\underline{10011}_{(2)} \times 2^{-4}_{(10)}$ représenté en simple précision, en décalant l'exposant de 127 par $2^{-4+127}_{(10)} = 2^{123}_{(10)} = 2^6 + 2^5 + 2^4 + 2^3 + 2 + 1$.
   Le même nombre est représenté en double précision, en décalant l'exposant de 1023 donc avec $2^{-4+1023}_{(10)} = 2^{1019}_{(10)} = 2^9 + 2^8 + 2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2 + 1$.On en déduit :

   - en simple précision : $\underbrace{01111011}_{8\text{ bits}}\ \underbrace{10011001100110011001101}_{23\text{ bits}}$ (le chiffre 1 final est dû à l'arrondi puisque le zéro initial aurait été suivi d'un 1)
   - en double précision $\underbrace{01111111011}_{11\text{ bits}}\ \underbrace{1001...1001}_{52\text{ bits}}$ (la séquence 1001 est répétée 13 fois dans la mantisse)

   L'erreur de représentation en simple précision est :
   $\Delta(x) = m(x) - x = 1,10011001100110011001101 \times 2^{-4}_{(10)} - 0,1 = 0,099999940395355225 - 0,1 \simeq 5,96.10^{-9}$

Exercice 2.5

Considérons un système de représentation binaire pour des nombres réels non négatifs avec trois bits pour l'exposant et trois bits pour la mantisse

1. Représenter les différents exposants

   SOLUTION :

   Le biais de ce système est égal à $B = 2^{q-1} - 1 = 2^2 - 1 = 3$. L'exposant maximal étant égal à 7, on dispose de 8 valeurs, dont deux sont réservées, et on a donc 3 valeurs d'exposant négatives ou nulles et trois positives.

Nombre codé	Nombre décimal équivalent	Valeur de l'exposant correspondant
000_2	0_{10}	$2^{0-B} = 2^{0-3} = 2^{-3}$
001_2	1_{10}	$2^{1-3} = 2^{-2}$
010_2	2_{10}	$2^{2-3} = 2^{-1}$
011_2	3_{10}	$2^{3-3} = 2^0 = 1$
100_2	4_{10}	$2^{4-3} = 2^1 = 2$
101_2	5_{10}	$2^{5-3} = 2^2 = 4$
110_2	6_{10}	$2^{6-3} = 2^3 = 8$
111_2	7_{10}	$2^{7-3} = 2^4 = 16$

2. Evaluer les valeurs binaires et leur équivalent en décimal de toutes les mantisses indépendamment de l'exposant et du bit caché.

   SOLUTION :

Nombre codé	Valeur décimale équivalente	Valeur décimale en tenant compte d'un bit caché nul
000_2	0_{10}	$0.000 = 0$
001_2	1_{10}	$0.001 = 2^{-3} = 0.125$
010_2	2_{10}	$0.010 = 2^{-2} = 0.25$
011_2	3_{10}	$0.011 = 2^{-2} + 2^{-3} = 0.375$
100_2	4_{10}	$0.100 = 2^{-1} = 0.5$
101_2	5_{10}	$0.101 = 2^{-1} + 2^{-3} = 0.625$
110_2	6_{10}	$0.110 = 2^{-1} + 2^{-2} = 0.75$
111_2	7_{10}	$0.111 = 2^{-1} + 2^{-2} + 2^{-3} = 0.875$

3. Calculer le plus petit et le plus grand nombres stockables.

   SOLUTION :

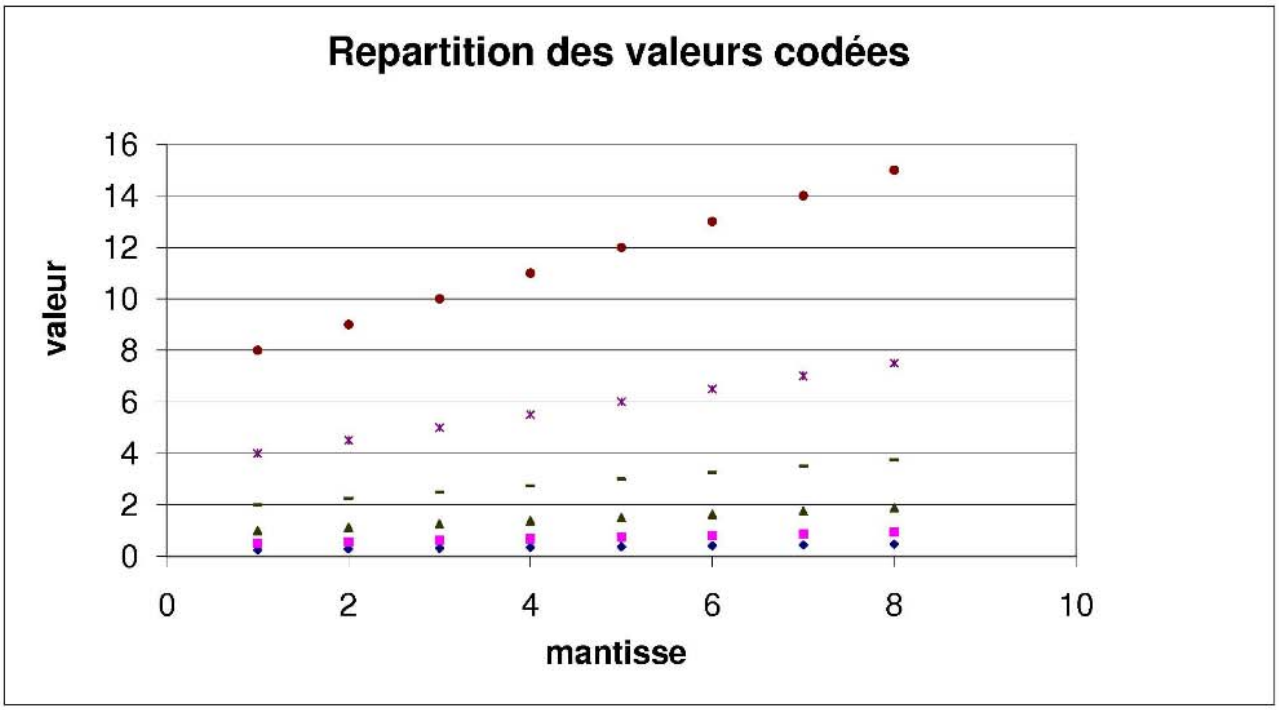
   - plus petit nombre stockable : exposant 001, mantisse 000, soit $1.000 \times 2_{10}^{-2} = 0.25$
   - plus grand nombre stockable : exposant 110, mantisse 111, soit $1.111 \times 2_{10}^{3} = \left(1 + 2^{-1} + 2^{-2} + 2^{-3}\right) \times 8 = 15$

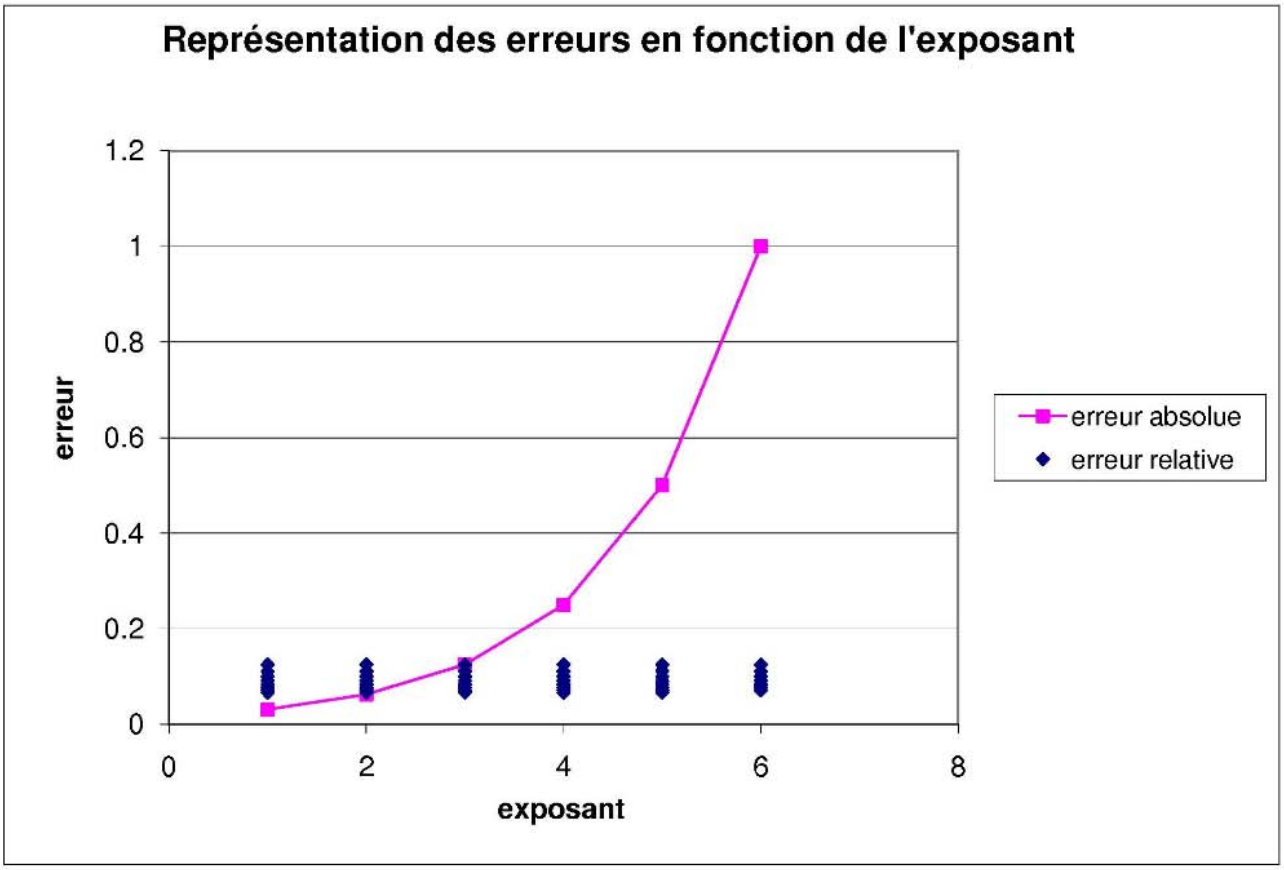
4. Evaluer la précision de la machine

   SOLUTION : eps $= 2^{-q} = 2^{-3} = 0.125$

5. Calculer l'intervalle de la droite des réels qui peut être représenté par ce système et faire un diagramme de la distribution des nombres normalisés flottants de cet intervalle. Commentaires.

   SOLUTION : Les nombres représentables sont compris entre 0.25 et 15.





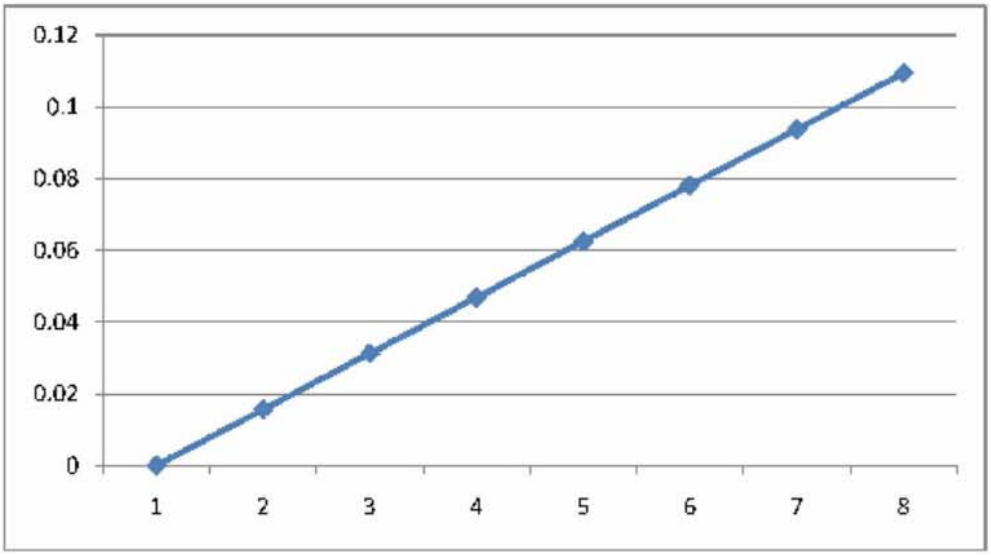
Comme le montre le second graphique : l'erreur relative entre deux valeurs consécutives est constante, par contre l'erreur absolue augmente beaucoup, et ce au fur et à mesure que l'exposant augmente.

6. Répétez pour les nombres sous-normalisés.

SOLUTION : Ce sont les nombres dont l'exposant est 000 et qui ne comportent pas de bit caché. Dans une machine à trois bits de mantisse, il en existe 2^3. Ils sont donc :

Nombre codé	Nombre complet	Nombre décimal équivalent	Valeur
000_2	0.000	$0_{10} \times 2^{-3}$	0
001_2	0.001	$2^{-3} \times 2^{-3}$	$2^{-6} = \frac{1}{64} = 1.5625 \times 10^{-2}$
010_2	0.010	$2^{-2} \times 2^{-3}$	$2^{-5} = 0.03125$
011_2	0.011	$\left(2^{-2} + 2^{-3}\right) \times 2^{-3}$	$2^{-6} + 2^{-5} = 4.6875 \times 10^{-2}$
100_2	0.100	$2^{-1} \times 2^{-3}$	$2^{-4} = 0.062\,5$
101_2	0.101	$\left(2^{-1} + 2^{-3}\right) \times 2^{-3}$	$2^{-4} + 2^{-6} = 7.8125 \times 10^{-2}$
110_2	0.110	$\left(2^{-1} + 2^{-2}\right) \times 2^{-3}$	$2^{-4} + 2^{-5} = 0.09375$
111_2	0.111	$\left(2^{-1} + 2^{-2} + 2^{-3}\right) \times 2^{-3}$	$2^{-4} + 2^{-5} + 2^{-6} = 0.10938$

Représentation :



L'erreur relative et l'erreur absolue entre deux valeurs consécutives sont constantes.

7. Quel est le nombre de valeurs flottantes différentes que nous pouvons représenter par ce système ? Généraliser.

   SOLUTION : Entre les valeurs inférieure l_- et supérieure l_+ des exposants on a $l_+ - l_- + 1$ valeurs, soit

   $$l_+ - l_- + 1 = B - (-B + 1) + 1 = 2B = 2(2^{q-1} - 1)$$

   valeurs. Si on a p chiffres de mantisse on a 2^p nombres possibles.

   Sans prendre en compte les valeurs non utiles, on a donc en tout $2^p.2(2^{q-1} - 1)$ valeurs, soit ici 96 valeurs non signées.

8. Comparer, en utilisant cet exemple, le système de représentation en virgule flottante avec celui en virgule fixe, et établir les avantgaes et inconvénients de chaque système.

   SOLUTION : En virgule fixe, l'écart entre deux valeurs est constant, donc plus la valeur est petite, plus l'erreur relative est grande.

Exercice 2.6

```
function [epsilon_machine]=chercher_epsilon_2()
puissance = 1;
while 1+2^(-puissance) <> 1
puissance = puissance +1;
end
epsilon_machine= 2^(-puissance+1);
printf(' \n valeur de puissance : %d ',puissance-1);
printf(' \n valeur de epsilon_machine : %e \n',epsilon_machine);
endfunction
```

Exercice 2.7

```
function [B,P,A]=determin_base_mantisse()
// Détermination de la base B
// et du nombre P de places de la mantisse
// de l'ordinateur
A = 1.0;
B = 1.0;
P = 0;
while ((A + 1.0) - A) - 1.0 == 0.0
```

```
P = P +1;
A = 2 * A;
end;
P=P-1;
// A est alors le plus grand nombre codable
// P est le nombre de positions sur lesquelles on met des bits dans la mantisse
// pour trouver B on cherche quel nombre on peut ajouter au plus grand nombre codable
// avant de dépasser la capacité de calcul
// si B est la base du système virgule flottante utilisé et p le nombre de chiffres de mantisse,
// on peut montrer qu'à la fin de la première boucle « tant que », A vérifie B^p <= A < B^(p+1).
while ((A + B) - A) - B <> 0.0
B = B + 1.0;
end;
printf(' A : %e, B : %e, P :%d \n',A,B,P);
endfunction
```

Exercice 2.8

Voici deux programmes qui calculent le plus petit nombre normalisé :

CODE 1

```
clear()
format("e",22);
x=1;
t=x;
while %eps*t/2 >0
t=%eps*t/2;
if (t>0) then
x=t;
end;
end;
printf("valeur de x : %22.20e",x);
```

CODE 2

```
clear()
format("e",22);
t=1;
while %eps*t >0
ta=t;
t=t/2;
end;
t=ta;
printf("valeur de t : %22.20e",t);
```

Pour resituer :

- ε le plus petit nombre positif normalisé tel que $1 + \varepsilon \neq 1$. On a vu au premier exercice que $\varepsilon = 2^{-p} = 2^{-52} = 2,22.10^{-16}$ où p est le nombre de chiffres de la mantisse.
- soit x le plus petit nombre positif normalisé, il correspond à un nombre de la forme suivante en décimal

$$x = 2^{1-B}$$

où B est le Biais donné par $B = 2^{q-1} - 1$ où q représente le nombre de bits de l'exposant. Donc $x = 2^{2-2^{1}} = 2^{1-1023} = 2^{-1022}$ en double précision. On a vu au premier exercice qu'en double précision, on obtient $1,\underbrace{000...0}_{52 \text{ fois}} \times 10^{00000001}_{(2)} = 2^{1-1023} = 2^{-1022} = 2,2250738.10^{-308}$.

Il est clair que le plus petit nombre codable est forcément plus petit que ε et que les deux s'expriment comme des puissances de 2, donc il est logique de rechercher x à partir de ε, comme un diviseur de ce dernier par exemple.

Quand on examine ce que font les deux codes à partir des premiers itérés, on constate que CODE 1 recherche x comme une puissance de ε. En particulier le x est de la forme $x_1 = \left(\frac{\varepsilon}{2}\right)^n$. Le CODE 2 quant à lui, recherche un diviseur de ε : $x_2 = \frac{\varepsilon}{2}$.

Il n'y a pas de raison pour que $2^{2-2^{-1}}$ soit une puissance de 2^{-p}, ce qui reviendrait à chercher α tel que

$$2^{2-2^{-1}} = \left(2^{-p}\right)^{\alpha}$$

soit, un α entier tel que

$$\alpha = -\frac{2 - 2^{q-1}}{p}$$

donc le CODE 1 ne peut pas donner la bonne solution.

Le CODE 2 cherche n tel que

$$\frac{\varepsilon}{2^n} = \frac{2^{-p}}{2^n} = 2^{2-2^{-1}}$$

soit

$$2^{-n-p} = 2^{2-2^{-1}} \Longleftrightarrow 2 - 2^{q-1} = -n - p \Longleftrightarrow n = -2 + 2^{q-1} - p$$

Vérifions : en double précision on a $n = -2 + 2^{q-1} - p = -2 + 2^{11-1} - 52 = 2^{10} - 54 = 970.0$

$$x_2 = \frac{\varepsilon}{2^n} = \frac{2^{-52}}{2^{970}} = 2^{-1022} = 2.2251 \times 10^{-308}$$

Exercice 3.1

Soient trois nombres réels positifs a, b, c avec $a > b > c$. On calcule leur somme de deux façons :

1. $S = (a + b) + c$
2. $S = a + (b + c)$

Quel résultat est le meilleur ?

SOLUTION :

Pour deux nombres quelconques a et b, la somme s est entachée d'une erreur d'entrée et d'une erreur de calcul. On a donc :

$$\eta(s) = \eta(a + b) = \eta^I(a + b) + \eta^C(a + b) \tag{1}$$

L'erreur d'entrée, inérente à la représentation du résultat de l'opération, se décompose en fonction des erreurs de même type commises sur les opérandes, à savoir :

$$\eta^I(a + b) = \frac{a}{a + b}\eta^I(a) + \frac{b}{a + b}\eta^I(b) \tag{2}$$

Donc (1) donne :

$$\begin{aligned} \eta(a + b) &= \eta^I(a + b) + \eta^C(a + b) \\ &= \frac{a}{a + b}\eta^I(a) + \frac{b}{a + b}\eta^I(b) + \eta^C(a + b) \end{aligned}$$

Si on utilise ce résultat pour la somme de trois réels, on obtient :

$$\begin{aligned} \eta\left((a + b) + c\right) &= \eta\left(s + c\right) = \eta^I(s + c) + \eta^C(s + c) \\ &= \frac{a + b}{a + b + c}\eta(a + b) + \frac{c}{a + b + c}\eta^I(c) + \eta^C(a + b + c) \end{aligned}$$

car le résultat de l'opération $a + b$ se récpercute au niveau de la valeur de s.

Or d'après (2)

$$\eta(a + b) = \frac{a}{a + b}\eta^I(a) + \frac{b}{a + b}\eta^I(b) + \eta^C(a + b)$$

donc

$$\eta\left((a + b) + c\right) = \frac{a + b}{a + b + c}\left[\frac{a}{a + b}\eta^I(a) + \frac{b}{a + b}\eta^I(b) + \eta^C(a + b)\right] + \frac{c}{a + b + c}\eta^I(c) + \eta^C(a + b + c)$$

d'où

$$\begin{aligned}\eta\left((a+b)+c\right) &= \frac{a}{a+b+c}\eta^I(a)+\frac{b}{a+b+c}\eta^I(b)+\frac{a+b}{a+b+c}\eta^C(a+b)+\frac{c}{a+b+c}\eta^I(c)+\eta^C(a+b+c)\\ &= \frac{1}{a+b+c}\left[a\eta^I(a)+b\eta^I(b)+c\eta^I(c)+(a+b)\eta^C(a+b)+(a+b+c)\eta^C(a+b+c)\right]\end{aligned}$$

Si on introduit

$$R_I = \max(\eta^I(a),\eta^I(b),\eta^I(c))$$

le maximum des erreurs d'entrée des trois constantes, et comme par ailleurs les erreurs de calculs sont majorées par l'epsilon machine :

$$\max\left\{\left|\eta^C(a+b)\right|,\left|\eta^C(a+b+c)\right|\right\} \leq eps$$

on obtient :

$$\left|\eta\left((a+b)+c\right)\right| \leq \frac{1}{a+b+c}\left[(a+b+c)R_I+(a+b)eps+(a+b+c)eps\right]$$

donc

$$\left|\eta\left((a+b)+c\right)\right| \leq R_I+\frac{2a+2b+c}{a+b+c}eps \tag{3}$$

Pour le calcul de l'erreur commise lorsqu'on calcule $S=a+(b+c)$ il suffit de faire une permutation circulaire :

$\begin{array}{ccc} a & b & c \\ \downarrow & \downarrow & \downarrow \\ b & c & a \end{array}$ ce qui conduit au résultat dédut de (3) suivant :

$$\left|\eta\left(a+(b+c)\right)\right| \leq R_I+\frac{2b+2c+a}{a+b+c}eps \tag{4}$$

Pour comparer les bornes supérieures, on doit donc comparer les quantités $2a+2b+c$ et $2b+2c+a$.
On a

$$\begin{aligned}2a+2b+c &< 2b+2c+a\\ &\Leftrightarrow 2a+c<2c+a\\ &\Leftrightarrow a<c\end{aligned}$$

Or on est dans le cas où $a>c$ donc $2a+2b+c>2b+2c+a$ donc la borne supérieure de $\left|\eta\left((a+b)+c\right)\right|$ est plus grande que celle de $\left|\eta\left(a+(b+c)\right)\right|$. On en déduit qu'il y a un risque que le calcul de $(a+b)+c$ produise une erreur plus grande que celui de $a+(b+c)$. En conclusion: lorsqu'on somme des nombres il faut commencer par les plus petits.