

Analyse Numérique I*

Sup'Galilée, Ingénieurs MACS, 1ère année / L3 MIM

François Cuvelier

Laboratoire d'Analyse Géométrie et Applications
Institut Galilée
Université Paris XIII.

2026/09/07

- Volume horaire : $11 \times 3\text{h}$ cours - $11 \times 3\text{h}$ TD - 3h TP (installation) ,
- Prérequis des cours : *Equations différentielles*, *Projets numériques*, *Elements Finis*, *Volumes Finis*, ...
- Note finale : $(P_1 + P_2)/2$ avec points de bonus
 - ▶ P_1 partiel 1, 16 novembre 2026 , (prévision, après 6 cours et 6 TDs)
 - ▶ P_2 partiel 2, 6 janvier 2027,
- 1 groupe de TD,
- Un serveur Discord dédié aux cours et aux TDs,
- Un **polycopié**,
- Une page web dédiée
 - ▶ **cours** : <https://www.math.univ-paris13.fr/~cuvelier>
- Mail: cuvelier@math.univ-paris13.fr.
- Pas présent sur l'ENT!

Outils de base de l'analyse numérique et du calcul scientifique.

- Mathématiques
- Numériques
- Algorithmique

[GGK14, Dem12, Huc, Cia06, Las04, QSS07]

[Cia06] P.G. Ciarlet, *Introduction à l'analyse numérique matricielle et à l'optimisation*, DUNOD, 2006.

[Dem12] J.P. Demailly, *Analyse numérique et équations différentielles*, Grenoble Sciences, EDP Sciences, 2012.

[GGK14] W. Gander, M.J. Gander, and F. Kwok, *Scientific computing : an introduction using maple and matlab*, Springer, Cham, 2014.

[Huc] T. Huckle, *Collection of software bugs*
<http://www.zenger.informatik.tu-muenchen.de/persons/huckle/bugse.html>.

[Las04] R. Lascaux, P. et Théodor, *Analyse numérique matricielle appliquée à l'art de l'ingénieur*, Analyse numérique matricielle appliquée à l'art de l'ingénieur, no. vol. 1 et 2, Dunod, 2004.

[QSS07] A. Quarteroni, R. Sacco, and F. Saleri, *Méthodes Numériques: Algorithmes, analyse et applications (French Edition)*, 1 ed., Springer, September 2007.

Chapitre 1 Erreurs : arrondis, bug and Co.

Chapitre 2 Langage algorithmique.

Chapitre 3 Rappels algèbre linéaire.

Chapitre 4 Résolution de systèmes non-linéaires.

Chapitre 5 Résolution de systèmes linéaires.

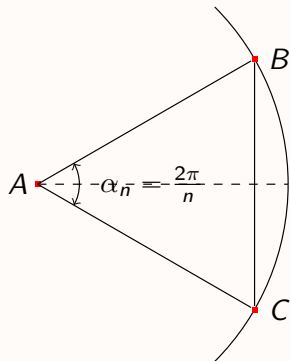
Chapitre 6 Polynômes d'interpolation.

Chapitre 7 Intégration numérique.

Chapitre I

Erreurs : arrondis, bug and Co.

Un exemple : calcul approché de π



- Aire du cercle : $\mathcal{A} = \pi r^2$.
- Archimède vers 250 avant J-C : $\pi \in]3 + \frac{10}{71}, 3 + \frac{1}{7}[$ avec 96 polygones.
($3.14084507042254 < \pi \approx 3.14159265358979 < 3.14285714285714$)
- Algorithme polygones inscrits ($r = 1$) :

$$\mathcal{A} = \lim_{n \rightarrow +\infty} \mathcal{A}_n = \lim_{n \rightarrow +\infty} \frac{n}{2} \sin(\alpha_n) \text{ et } \sin \frac{\alpha_n}{2} = \sqrt{\frac{1 - \sqrt{1 - \sin^2 \alpha_n}}{2}}.$$

```
1:  $s \leftarrow 1, n \leftarrow 4$ 
2: Tantque  $s > 1e - 10$  faire
3:    $s \leftarrow \text{sqrt}((1 - \text{sqrt}(1 - s * s))/2)$ 
4:    $n \leftarrow 2 * n$ 
5:    $A \leftarrow (n/2) * s$ 
6: Fin Tantque
```

▷ Initialisations
▷ Arrêt si $s = \sin(\alpha)$ est petit
▷ nouvelle valeur de $\sin(\alpha/2)$
▷ nouvelle valeur de n
▷ nouvelle valeur de l'aire du polygone

Un exemple : calcul approché de π

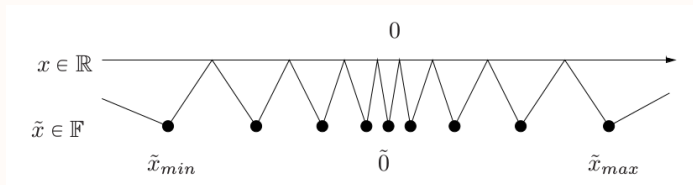
n	$\mathcal{A}_n$	$ \mathcal{A}_n - \pi $	$\sin(\alpha_n)$
4	2.000000000000000	1.141593e+00	1.000000e+00
64	3.13654849054594	5.044163e-03	9.801714e-02
4096	3.14159142150464	1.232085e-06	1.533980e-03
32768	3.14159263346325	2.012654e-08	1.917476e-04
65536	3.14159265480759	1.217796e-09	9.587380e-05
131072	3.14159264532122	8.268578e-09	4.793690e-05
524288	3.14159291093967	2.573499e-07	1.198423e-05
4194304	3.14159655370482	3.900115e-06	1.498030e-06
67108864	3.14245127249413	8.586189e-04	9.365235e-08
2147483648	0.000000000000000	3.141593e+00	0.000000e+00

Nombres flottants en machine

$$\begin{aligned}\tilde{x} &= \pm m \cdot b^e \\ m &= D.D \cdots D, \quad \text{mantisse} \\ e &= D \cdots D, \quad \text{exposant}\end{aligned}$$

où $D \in \{0, 1, \dots, b-1\}$ représente un chiffre et b la base.

Mantisse normalisée : 1er D (avant point) est non nul.



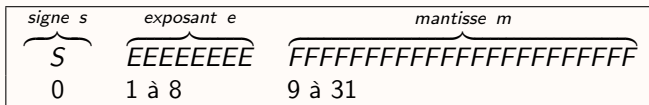
- **Précision machine** : plus petit nombre machine $\text{eps} > 0$ tel que

$$1 + \text{eps} > 1.$$

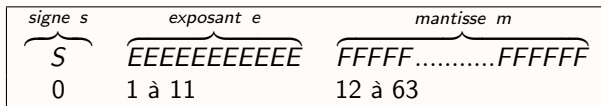
Matlab/Octave, langage C (double), ... : $\text{eps} = 2.2204460492503131e-16$

Système IEEE 754 (1985)

- **simple précision** : format 32 bits (4 octets), float en langage C.



- **double précision** : format 64 bits (8 octets), double en langage C.



dimension tableau	type	dimension (octets)	dimension total
1000×1000	float	4	$4 * 1000 * 1000 = 4Mo$
$10^4 \times 10^4$	float	4	$4 * 10^8 = 400Mo$
1000×1000	double	8	$8 * 1000 * 1000 = 8Mo$
$10^4 \times 10^4$	double	8	$8 * 10^8 = 800Mo$
$10^5 \times 10^5$	double	8	$8 * 10^{10} = 80Go$

En langage C (par ex.):

- `int a=2147483647,b; alors b=a+1; donne b=-2147483648,`
- `double a=1/2,b;b=a+1; donne a==0 et b==1,`

En Matlab (par ex.) :

- `x=eps; (x > 0) alors 1+x==1 est faux (false=0)`
- `x=eps/2;y=1; alors (y+x)+x==1 est vrai et y+(x+x)==1 est faux.`
`(x + y) + z = x + (y + z)` n'est pas forcément vérifiée sur machine!

Il faut être attentifs aux signes sur machine.

- $f(x) = \frac{1}{1-\sqrt{1-x^2}}$, si $|x| \leq 0.5\sqrt{\text{eps}}$ alors $\sqrt{1-x^2} \equiv 1$!

$$\Rightarrow \bar{x} = 0.5\sqrt{\text{eps}}, \quad \boxed{\frac{1}{1-\sqrt{1-\bar{x}^2}} \equiv +\infty}$$

•

$$f(x) = \frac{1}{1-\sqrt{1-x^2}} = \frac{1}{1-\sqrt{1-x^2}} \times \frac{1+\sqrt{1-x^2}}{1+\sqrt{1-x^2}} = \frac{1+\sqrt{1-x^2}}{x^2}$$

$$\Rightarrow \bar{x} = 0.5\sqrt{\text{eps}}, \quad \boxed{\frac{1+\sqrt{1-\bar{x}^2}}{\bar{x}^2} \equiv 3.6029e+16}$$

Erreurs d'annulation : calcul de π , le retour

$$\sin \frac{\alpha_n}{2} = \sqrt{\frac{1 - \cos \alpha_n}{2}} = \sqrt{\frac{1 - \sqrt{1 - \sin^2 \alpha_n}}{2}},$$

C'est le calcul de $1 - \sqrt{1 - \sin^2 \alpha_n}$ qui pose problème!

$$1 - \cos \alpha_n = (1 - \cos \alpha_n) \frac{1 + \cos \alpha_n}{1 + \cos \alpha_n} = \frac{\sin^2 \alpha_n}{1 + \cos \alpha_n} \Rightarrow \sin \frac{\alpha_n}{2} = \frac{\sin \alpha_n}{\sqrt{2(1 + \sqrt{1 - \sin^2 \alpha_n})}}$$

```
1:  $s \leftarrow 1, n \leftarrow 4,$   
2: Tantque  $s > 1e - 10$  faire  
3:    $s \leftarrow s / \text{sqrt}(2 * (1 + \text{sqrt}(1 - s * s)))$   
4:    $n \leftarrow 2 * n$   
5:    $A \leftarrow (n/2) * s$   
6: Fin Tantque
```

▷ Initialisations

▷ Arrêt si $s = \sin(\alpha)$ est petit

▷ nouvelle valeur de $\sin(\alpha/2)$

▷ nouvelle valeur de n

▷ nouvelle valeur de l'aire du polygone

Un exemple : calcul approché de π

n	A_n	$ A_n - \pi $	$\sin(\alpha_n)$
4	2.000000000000000	1.141593e+00	1.000000e+00
64	3.13654849054594	5.044163e-03	9.801714e-02
4096	3.14159142151120	1.232079e-06	1.533980e-03
32768	3.14159263433856	1.925123e-08	1.917476e-04
65536	3.14159264877699	4.812807e-09	9.587380e-05
131072	3.14159265238659	1.203202e-09	4.793690e-05
524288	3.14159265351459	7.519985e-11	1.198422e-05
4194304	3.14159265358862	1.174172e-12	1.498028e-06
67108864	3.14159265358979	3.552714e-15	9.362676e-08
2147483648	3.14159265358979	1.332268e-15	2.925836e-09

Mais avant de poursuivre ...



(a) Pont de la Basse-Chaine, Angers (1850)



(b) Takoma Narrows Bridge, Washington (1940)



(c) Millenium Bridge, London (2000)

Figure: Une histoire de ponts

Chapitre II

Langage algorithmique

♥ Definition 1.1 : *Petit Robert 97*

Algorithmique : Enchaînement d'actions nécessaires à l'accomplissement d'une tâche.

Exemple Permutation

Nous voulons permutter deux voitures sur un parking de trois places numérotées de 1 à 3 et ceci sans gêner la circulation.

La première voiture, une Renault Zoé, est sur l'emplacement 2, la seconde, une Citroën C1, est sur l'emplacement 3.

Donner un algorithme permettant de résoudre cette tâche.

Exemple Résolution

Donner un algorithme permettant de résoudre

$$ax = b$$

Caractéristiques d'un *bon* algorithme:

- Il ne souffre d'aucune ambiguïté $\Rightarrow$ très clair.
- Combinaison d'opérations (actions) élémentaires.
- Pour toutes les données d'entrée, l'algorithme doit fournir un résultat en un nombre fini d'opérations.

Première approche méthodologique:

- ① Définir clairement le problème.
- ② Rechercher une méthode de résolution (formules, ...).
- ③ Ecrire l'algorithme (par raffinement successif pour des algorithmes *compliqués*).

- 1 Introduction
- 2 Pseudo-langage algorithmique

- Les bases
- Les tableaux, vecteurs, matrices

Vocabulaire de base:

- ① constantes, variables,
- ② opérateurs (arithmétiques, relationnels, logiques),
- ③ expressions,
- ④ instructions (simples et composées),
- ⑤ fonctions.

① Constantes, variables:

- ▶ Constante $\Rightarrow$ symbole, identificateur non modifiable
- ▶ Une variable est un objet dont la valeur est modifiable, qui possède un nom et un type (entier, caractère, réel, complexe, tableau, matrice, vecteur...).
- ▶ Donnée $\Rightarrow$ introduite par l'utilisateur (souvent sous forme d'une variable)

2 Opérateurs:

► Opérateurs arithmétiques:

Nom	Symbole	Exemple
addition	+	$a + b$
soustraction	-	$a - b$
opposé	-	$-a$
produit	*	$a * b$
division	/	a / b

► Opérateurs logiques:

Nom	Symbole	Exemple
négation	~	$\sim a$
ou		$a b$
et	&	$a \& b$

► Opérateurs relationnels:

Nom	Symbole	Exemple
identique	==	$a == b$
différent	~=	$a \sim = b$
inférieur	<	$a < b$
supérieur	>	$a > b$
inférieur ou égal	<=	$a < = b$
supérieur ou égal	>=	$a > = b$

► Opérateur d'affectation:

Nom	Symbole	Exemple
affectation	←	$a \leftarrow b$

- ③ **Expression:** groupe d'opérandes (i.e. nombres, constantes, variables, ...) liées par certains opérateurs pour former un terme algébrique qui représente une valeur (i.e. un élément de donnée simple)

👉 **Exemple d'expression numérique:**

$$(b * b - 4 * a * c) / (2 * a)$$

Opérandes: identifiants a , b , c , constantes 4 et 2.
--

Opérateurs: symboles $*$, $-$ et $/$
--

👉 **Exemple d'expression booléenne:**

$$(x < 3.14)$$

Opérandes: identifiants x et contante 3.14

Opérateurs: symboles $<$

- ④ **Instructions:** ordre ou un groupe d'ordres qui déclenche l'exécution de certaines actions par l'ordinateur. Il y a deux types d'instructions : simples et structurées.

👉 **Instructions simples:**

- ★ affectation d'une valeur a une variable.
- ★ appel d'une fonction (procedure, subroutine, ... suivant les langages).

👉 **Instructions structurées:**

- ★ les instructions composées, groupe de plusieurs instructions simples,
- ★ les instructions répétitives, permettant l'exécution répétée d'instructions simples, (i.e. boucles «pour», «tant que»)
- ★ les instructions conditionnelles, lesquels ne sont exécutées que si une certaine condition est respectée (i.e. «si»)

Exemple : boucle «pour»

Algorithme Exemple boucle «pour»

Données : n , un entier positif

- 1: $S \leftarrow 0$
 - 2: **Pour** $i \leftarrow 1$ à n **faire**
 - 3: $S \leftarrow S + \cos(i^2)$
 - 4: **Fin Pour**
-

Listing: (Matlab) Exemple boucle for

```
n=input('n=');           1
assert(n>=0)              2
S=0;                      3
for i=1:n                 4
    S=S+cos(i^2);         5
end                       6
```

Mais que fait-il?

$S = ?$

Exemple : boucle «tant que»

Algorithme Exemple boucle «tant que»

```
1:  $i \leftarrow 0, x \leftarrow 1$   
2: Tantque  $i < 1000$  faire  
3:    $x \leftarrow x + i * i$   
4:    $i \leftarrow i + 1$   
5: Fin Tantque
```

Listing: (Matlab) Exemple boucle while

```
i=0;x=1; 1  
while (i<1000) 2  
    x=x+i*i; 3  
    i=i+1; 4  
end 5
```

Mais que fait-il?

$i = ?$, $x = ?$

Exemple : instructions conditionnelles «si»

Algorithme Exemple instruction «si»

Données : *age*, un entier.

- 1: **Si** *age* $\geq$ 64 **alors**
 - 2: *affiche*('retraite')
 - 3: **Sinon Si** *age* $\geq$ 18 **alors**
 - 4: *affiche*('majeur')
 - 5: **Sinon**
 - 6: *affiche*('mineur')
 - 7: **Fin Si**
-

Listing: (Matlab) Exemple instruction if

```
age=input('age=');           1
if age>=64                    2
    disp('retraite')          3
elseif age>=18                4
    disp('majeur')            5
else                           6
    disp('mineur')            7
end                             8
```

- 1 Introduction
- 2 Pseudo-langage algorithmique
 - Les bases
 - Les tableaux, vecteurs, matrices

Pour simplifier l'écriture des algorithmes

- Identification des tableaux 1D à des vecteurs:

- ▶ $\mathbf{x}$ tableau de réels à une dimension et de longueur $n \in \mathbb{N}^* \Rightarrow \mathbf{x} \in \mathbb{R}^n$,
- ▶ $\mathbf{x}$ tableau d'entier à une dimension et de longueur $n \in \mathbb{N}^* \Rightarrow \mathbf{x} \in \mathbb{N}^n$,
- ▶ ...

Choix indiciage: l'indice du premier élément d'un tableau/vecteur est 1

$$\forall i \in \llbracket 1, n \rrbracket, \mathbf{x}_i \text{ (not. mathématique)} \iff \mathbf{x}(i) \text{ (not. algorithmique)}$$

Soit $\mathbf{x} \in \mathbb{R}^n$ donné $\implies n \in \mathbb{N}^*$ est implicitement donné:

$$n \leftarrow \text{length}(\mathbf{x}), \text{ par exemple.}$$

- Identification des tableaux 2D à des matrices:

- ▶ $\mathbb{T}$ tableau de réels à m lignes et n colonnes $\Rightarrow \mathbb{T} \in \mathcal{M}_{m,n}(\mathbb{R})$,
- ▶ $\mathbb{T}$ tableau de complexes à m lignes et n colonnes $\Rightarrow \mathbb{T} \in \mathcal{M}_{m,n}(\mathbb{C})$,
- ▶ ...

$$\forall (i, j) \in \llbracket 1, m \rrbracket \times \llbracket 1, n \rrbracket, \mathbb{T}_{i,j} \text{ (not. mathématique)} \iff \mathbb{T}(i, j) \text{ (not. algorithmique)}$$

Soit $\mathbb{T} \in \mathcal{M}_{m,n}(\mathbb{R})$ donné $\implies m, n \in \mathbb{N}^*$ sont implicitement donnés:

$$[m, n] \leftarrow \text{size}(\mathbb{T}), \text{ par exemple.}$$

Exercice 1



Soient $(a, b) \in \mathbb{R}^2$, $a < b$, $N \in \mathbb{N}$ et $\forall i \in \llbracket 0, N \rrbracket$, $x_i = a + i \frac{b-a}{N}$, ensemble des points de la discrétisation régulière de $[a, b]$ avec N pas (ou $N + 1$ points).

Q. 1

Écrire une fonction **DisReg** permettant de retourner l'ensemble des $(x_i)_{i=0}^N$.

Soient $t^0 = 1$, $T = 10$ et $N = 100$.

Q. 2

En utilisant la fonction **DisReg**, écrire un programme permettant d'obtenir l'ensemble des $(t^n)_{n=0}^N$ correspondant à la discrétisation régulière de l'intervalle $[t^0, t^0 + T]$ avec N pas.