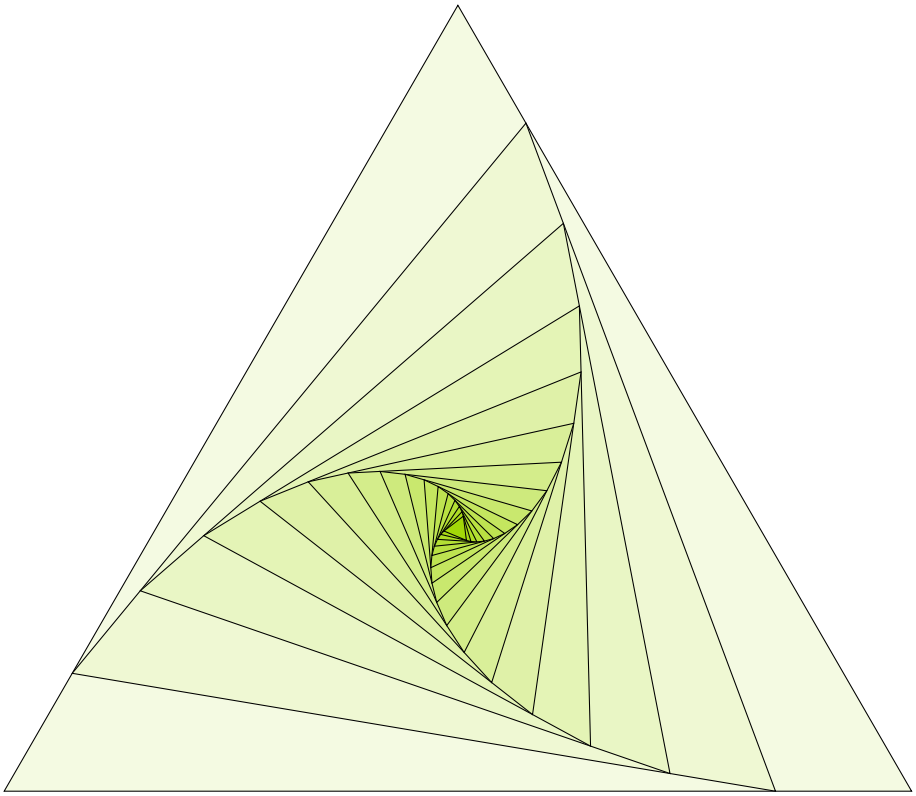
Analyse numérique élémentaire

Notes de cours

Sup Galilée, Ingénieurs MACS 1ère année



Francois Cuvelier
Université Paris XIII / Institut Galilée
L.A.G.A./Département de Mathématiques

Table des matières

Chapitre 1

Représentation des nombres en machine, erreurs d'arrondis

 Toute cette partie est le contenu quasi-intégral d'un document réalisé par C. Japhet

Ce chapitre est une introduction à la représentation des nombres en machine et aux erreurs d'arrondis, basé sur [2], [1].

1.1 Un exemple : calcul approché de π

Cet exemple est extrait de [2], [1]. Le nombre π est connu depuis l'antiquité, en tant que méthode de calcul du périmètre du cercle ou de l'aire du disque. Le problème de la quadrature du cercle étudié par les anciens Grecs consiste à construire un carré de même aire qu'un cercle donné à l'aide d'une règle et d'un compas. Ce problème resta insoluble jusqu'au $19^{\text{ème}}$ siècle, où la démonstration de la transcendance de π montra que le problème ne peut être résolu en utilisant une règle et un compas.

Nous savons aujourd'hui que l'aire d'un cercle de rayon r est $\mathcal{A} = \pi r^2$. Parmi les solutions proposées pour approcher $\mathcal{A}$, une méthode consiste à construire un polygône dont le nombre de côté augmenterai jusqu'à ce qu'il devienne équivalent au cercle circonscrit. C'est Archimède vers 250 avant J-C qui appliquera cette propriété au calcul des décimales du nombre π, en utilisant à la fois un polygône inscrit et circonscrit au cercle. Il utilise ainsi un algorithme pour le calcul et parvient à l'approximation de π dans l'intervalle $(3 + \frac{1}{7}, 3 + \frac{10}{71})$ en faisant tendre le nombre de côtés jusqu'à 96.

Regardons l'algorithme de calcul par les polygônes inscrits. On considère un cercle de rayon $r = 1$ et on note $\mathcal{A}_n$ l'aire associée au polygône inscrit à n côtés. En notant $\alpha_n = \frac{2\pi}{n}$, $\mathcal{A}_n$ est égale à n fois l'aire du triangle ABC représenté sur la figure 1.1, c'est-à-dire

$$\mathcal{A}_n = n \cos\frac{\alpha_n}{2} \sin\frac{\alpha_n}{2},$$

que l'on peut réécrire

$$\mathcal{A}_n = \frac{n}{2}(2\cos\frac{\alpha_n}{2}\sin\frac{\alpha_n}{2}) = \frac{n}{2}\sin\alpha_n = \frac{n}{2}\sin(\frac{2\pi}{n}).$$

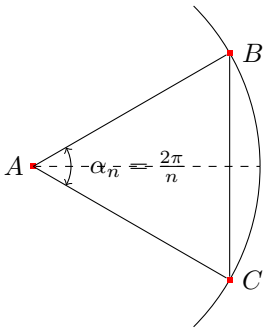


Figure 1.1: Quadrature du cercle {fig1}

Comme on cherche à calculer π à l'aide de A_n, on ne peut pas utiliser l'expression ci-dessus pour calculer A_n, mais on peut exprimer A_{2n} en fonction de A_n en utilisant la relation

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

Ainsi, en prenant $n = 2^k$, on définit l'approximation de π par récurrence

$$x_k = A_{2^k} = \frac{2^k}{2}s_k, \quad \text{avec } s_k = \sin(\frac{2\pi}{2^k}) = \sqrt{\frac{1-\sqrt{1-s_{k-1}}}{2}}$$

En partant de $k = 2$ (i.e. $n = 4$ et $s = 1$) on obtient l'algorithme suivant:

Algorithme 1.1 Algorithme de calcul de π, version naïve {AlgoPiNaif}

$s \leftarrow 1$, $n \leftarrow 4$ ▷ Initialisations
Tantque $s > 1e-10$ **faire** ▷ Arrêt si $s = \sin(\alpha)$ est petit
  $s \leftarrow sqrt((1 - sqrt(1 - s * s))/2$ ▷ nouvelle valeur de $\sin(\alpha/2)$
  $n \leftarrow 2 * n$ ▷ nouvelle valeur de n
  $A \leftarrow (n/2) * s$ ▷ nouvelle valeur de l'aire du polygône
Fin Tantque

On a $\lim_{k\longrightarrow+\infty} x_k = \pi$. Ce n'est pourtant pas du tout ce que l'on va observer sur machine! Les résultats en Python (sous Sage) de la table 1.1 montre que l'algorithme commence par converger vers π puis pour $n > 65536$, l'erreur augmente et finalement on obtient $A_n = 0$!! "Although the theory and the program are correct, we obtain incorrect answers" ([2]).

Ceci résulte du codage des valeurs réelles sur un nombre fini de bits, ce que nous allons détailler dans ce chapitre.

1.2 Représentation scientifique des nombres dans différentes bases

Dans cette section nous introduisons les notions de mantisse, exposant, et la façon dont sont représentés les nombres sur une calculatrice ou un ordinateur.

1.2.1 Partie entière, mantisse et exposant

Exemple en base 10

La base 10 est la base naturelle avec laquelle on travaille et celle que l'on retrouve dans les calculatrices.

Un nombre à virgule, ou nombre décimal, à plusieurs écritures différentes en changeant simplement la position du point décimal et en rajoutant à la fin une puissance de 10 dans l'écriture de ce nombre. La partie à gauche du point décimal est la partie entière, celle à droite avant l'exposant s'appelle la mantisse. Par exemple le nombre $x = 1234.5678$ à plusieurs représentations :

$$x = 1234.5678 = 1234.5678\ 10^0 = 1.2345678\ 10^3 = 0.0012345678\ 10^6, \tag{1.1}$$ {eqx}

n	A_n	$\lvert A_n - \pi \rvert$	$\sin(\alpha_n)$
4	2.00000000000000	1.141593e+00	1.000000e+00
8	2.82842712474619	3.131655e-01	7.071068e-01
16	3.06146745892072	8.012519e-02	3.826834e-01
32	3.12144515225805	2.014750e-02	1.950903e-01
64	3.13654849054594	5.044163e-03	9.801714e-02
128	3.14033115695474	1.261497e-03	4.906767e-02
256	3.14127725093276	3.154027e-04	2.454123e-02
512	3.14151380114415	7.885245e-05	1.227154e-02
1024	3.14157294036788	1.971322e-05	6.135885e-03
2048	3.14158772527996	4.928310e-06	3.067957e-03
4096	3.14159142150464	1.232085e-06	1.533980e-03
8192	3.14159234561108	3.079787e-07	7.669903e-04
16384	3.14159257654500	7.704479e-08	3.834952e-04
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
262144	3.14159260737572	4.621407e-08	2.396845e-05
524288	3.14159291093967	2.573499e-07	1.198423e-05
1048576	3.14159412519519	1.471605e-06	5.992115e-06
2097152	3.14159655370482	3.900115e-06	2.996060e-06
4194304	3.14159655370482	3.900115e-06	1.498030e-06
8388608	3.14167426502176	8.161143e-05	7.490335e-07
16777216	3.14182968188920	2.370283e-04	3.745353e-07
33554432	3.14245127249413	8.586189e-04	1.873047e-07
67108864	3.14245127249413	8.586189e-04	9.365235e-08
134217728	3.16227766016838	2.068501e-02	4.712161e-08
268435456	3.16227766016838	2.068501e-02	2.356080e-08
536870912	3.46410161513775	3.225090e-01	1.290478e-08
1073741824	4.00000000000000	8.584073e-01	7.450581e-09
2147483648	0.000000000000000	3.141593e+00	0.000000e+00

{tab1}

Table 1.1: Calcul de π avec l'algorithme naïf 1.1

avec

- *Partie entière* : 1234
- *Mantisse* : 0.5678 ou 1.2345678 ou 0.0012345678
- *Exposant* : 4 ou 6

Selon le décalage et l'exposant que l'on aura choisi, le couple mantisse-exposant va changer mais le nombre représenté est le même. Afin d'avoir une représentation unique, celle qui sera utilisée c'est la troisième dans (1.1), où la mantisse est 1.2345678 et l'exposant 3. C'est celle où le premier chiffre avant le point décimal dans la mantisse est non nul.

Exemple en base 2

C'est la base que les ordinateurs utilisent. Les chiffres utilisables en base 2 sont 0 et 1 que l'on appelle *bit* pour *binary digit*, les ordinateurs travaillent en binaire. Par exemple

$$39 = 32 + 4 + 2 + 1 = 2^5 + 2^2 + 2^1 + 2^0 = (100111)_2,$$
$$3.625 = 2^1 + 2^0 + 2^{-1} + 2^{-3} = (11.101)_2 = (1.1101)_2\ 2^1$$

Représentation d'un nombre en machine : nombres flottants

De façon générale tout nombre réels x sera représenté dans une base b ($b = 10$ pour une calculatrice $b = 2$ pour un ordinateur) par son signe (+ ou −), la mantisse m (appelée aussi significande) , la base

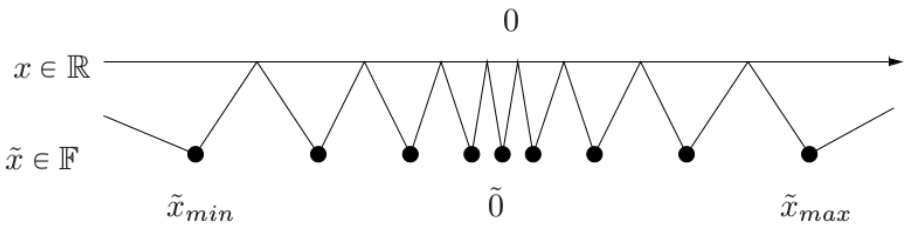
b et un exposant e tel que le couple (m, e) caractérise le nombre. En faisant varier e, on fait « flotter » la virgule décimale. La limitation fondamentale est que la place mémoire d'un ordinateur est limitée, c'est-à-dire qu'il ne pourra stocker qu'un ensemble fini de nombres. Ainsi un nombre machine réel ou *nombre à virgule flottante* s'écrira :

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

où $D \in \{0, 1, ..., b-1\}$ représente un chiffre. Des représentations approchées de π sont : (0.031,2), (3.142,0), (0.003,3) et on observe qu'elles ne donnent pas la même précision. Pour rendre la représentation unique et garder la meilleure précision, on utilisera une mantisse *normalisée* : le premier chiffre avant le point décimal dans la mantisse est non nul. Les nombres machine correspondants sont appelés *normalisés*. En base 2, le premier bit dans la mantisse sera donc toujours 1, et on n'écrit pas ce 1 ce qui permet d'économiser un bit. L'exposant est un nombre variant dans un intervalle fini de valeurs admissibles : $L \leqslant e \leqslant U$ (typiquement $L < 0$ et $U > 0$). Le système est donc caractérisé par quatre entiers :

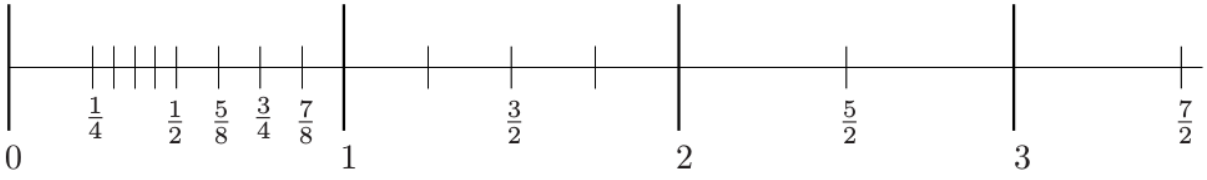
- la base b ($b = 2$),
- le nombre de chiffres t dans la mantisse (en base b),
- l'exposant minimal L et maximal U.

En mathématiques on effectue les calculs avec des nombres réels x provenant de l'intervalle continu $x \in [-\infty, \infty]$. A cause de la limitation ci-dessus, la plupart des réels seront approchés sur un ordinateur. Par exemple, $\frac{1}{3}, \sqrt{2}, \pi$ possèdent une infinité de décimales et ne peuvent donc pas avoir de représentation exacte en machine. Le plus simple des calculs devient alors approché. L'expérience pratique montre que cette quantité limitée de nombres représentables est largement suffisante pour les calculs. Sur l'ordinateur, les nombres utilisés lors des calculs sont des nombres machine $\tilde{x}$ provenant d'un ensemble discret de nombres machine $\tilde{x} \in \{\tilde{x}_{min}, \ldots, \tilde{x}_{max}\}$. Ainsi, chaque nombre réel x doit être transformé en un nombre machine $\tilde{x}$ afin de pouvoir être utilisé sur un ordinateur. Un exemple est donné sur la figure 1.1.



{fig2} Figure 1.2: Représentation des nombres réels $\mathbb{R}$ par les nombres machine $\mathbb{F}$

Prenons un exemple, beaucoup trop simple pour être utilisé mais pour fixer les idées: $t = 3$, $L = -1$, $U = 2$. Dans ce cas on a 3 chiffres significatifs et 33 nombres dans le système $\mathbb{F}$. Ils se répartissent avec 0 d'une part, 16 nombres négatifs que l'on ne représente pas ici, et 16 nombres positifs représentés sur la figure 1.3.



{fig3} Figure 1.3: Nombres positifs de $\mathbb{F}$ dans le cas $t = 3,\ L = -1,\ U = 2$

Dans cet exemple, l'écriture en binaire des nombres entre $\frac{1}{2}$ et 1 est

$$\frac{1}{2} = (0.100)_2, \quad \frac{3}{4} = \frac{1}{2} + \frac{1}{4} = (0.110)_2,$$
$$\frac{5}{8} = \frac{1}{2} + \frac{1}{8} = (0.101)_2, \quad \frac{7}{8} = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} = (0.111)_2.$$

On obtient ensuite les autres nombres en multipliant par une puissance de 2. Le plus grand nombre représentable dans ce système est $\frac{7}{2}$ (en particulier 4 n'est pas représentable). On remarque que les nombres ne sont pas espacés régulièrement. Ils sont beaucoup plus resserrés du côté de 0 entre $\frac{1}{4}$ et $\frac{1}{2}$ que entre 1 et 2 et encore plus qu'entre 2 et 3. Plus précisément, chaque fois que l'on passe par une puissance de 2, l'espacement absolu est multiplié par 2, mais l'espacement relatif reste constant ce qui est une bonne chose pour un calcul d'ingéniérie car on a besoin d'une précision absolue beaucoup plus grande pour des nombres petits (autour de un millième par exemple) que des nombres très grands (de l'ordre du million par exemple). Mais la précision ou l'erreur relative sera du même ordre. L'erreur absolue ou relative est définie à section 1.4.1.

Précision machine. elle est décrite par le nombre machine *eps*. *eps* est le plus petit nombre machine positif tel que $1 + eps > 1$ sur la machine. C'est la distance entre l'entier 1 et le nombre machine $\tilde{x} \in \mathbb{F}$ le plus proche, qui lui est supérieur. Dans l'exemple précédent $eps = 1/4$.

Sur une calculatrice

Le système utilisé est la base 10 ($b = 10$). Typiquement, il y a 10 chiffres pour la mantisse et 2 pour l'exposant ($L = -99$ et $U = 99$).

- Le plus grand nombre machine

$$\tilde{x}_{max} = 9.999999999 \times 10^{+99}$$

- Le plus petit nombre machine

$$\tilde{x}_{min} = -9.999999999 \times 10^{+99}$$

- Le plus petit nombre machine strictement positif

$$\tilde{x}_{+} = 1.000000000 \times 10^{-99}$$

  Notez qu'avec des nombres dénormalisés, ce nombre serait $0.000000001 \times 10^{-99}$, c'est-à-dire avec seulement un chiffre significatif!

Les différences de représentation des nombres flottants d'un ordinateur à un autre obligeaient à reprendre les programmes de calcul scientifique pour les porter d'une machine à une autre. Pour assurer la compatibilité entre les machines, depuis 1985 une norme a été proposée par l'IEEE (Institute of Electrical and Electronics Engineers), c'est la norme 754.

1.3 Nombres flottants : le système IEEE 754

Le système IEEE 754 est un standard pour la représentation des nombres à virgule flottante en binaire. Il définit les formats de représentation des nombres à virgule flottante (signe, mantisse, exposant, nombres dénormalisés) et valeurs spéciales (infinis et NaN). Le bit de poids fort est le bit de signe. Cela signifie que si ce bit est à 1, le nombre est négatif, et s'il est à 0, le nombre est positif. Les N_e bits suivants représentent l'exposant décalé, et les N_m bits suivants représentent la mantisse.

L'exposant est décalé de $2^{N_e-1} - 1$ (N_e représente le nombre de bits de l'exposant), afin de le stocker sous forme d'un nombre non signé.

1.3.1 simple précision

C'est le format 32 bits : 1 bit de signe, $N_e = 8$ bits d'exposant (-126 à 127), 23 bits de mantisse comme sur le tableau 1.2. L'exposant est décalé de $2^{N_e-1} - 1 = 2^7 - 1 = 127$.

signe s	*exposant e*	*mantisse m*
$\overbrace{S}$	$\overbrace{EEEEEEEE}$	$\overbrace{FFFFFFFFFFFFFFFFFFFFFFF}$
0	1 8	9 31

Table 1.2: Représentation en simple précision tab2

$\tilde{x}$	exposant e	mantisse m
$\tilde{x} = 0$ (si $S = 0$) $\tilde{x} = -0$ (si $S = 1$)	$e = 0$	$m = 0$
Nombre *normalisé* $\tilde{x} = (-1)^S \times 2^{e-127} \times 1.m$	$0 < e < 255$	quelconque
Nombre *dénormalisé* $\tilde{x} = (-1)^S \times 2^{e-126} \times 0.m$	$e = 0$	$m \neq 0$
$\tilde{x} =$ `Inf` (si $S = 0$) $\tilde{x} = -$`Inf` (si $S = 1$)	$e = 255$	$m = 0$
$\tilde{x} =$ `NaN` *(Not a Number)*	$e = 255$	$m \neq 0$

Table 1.3: Représentation en simple précision tab3

- Le *plus petit nombre positif normalisé* différent de zéro, et le *plus grand nombre négatif normalisé* différent de zéro sont :
  $\pm 2^{-126} = \pm 1,175494351 \times 10^{-38}$
- Le *plus grand nombre positif fini*, et le *plus petit nombre négatif fini* sont : $\pm(2^{24} - 1) \times 2^{104} = \pm 3,4028235 \times 10^{38}$

$\tilde{x}$	signe	exposant e	mantisse m	valeur
zéro	0	0000 0000	000 0000 0000 0000 0000 0000	0,0
	1	0000 0000	000 0000 0000 0000 0000 0000	-0,0
1	0	0111 1111	000 0000 0000 0000 0000 0000	1,0
Plus grand nombre normalisé	0	1111 1110	111 1111 1111 1111 1111 1111	$3,4 \times 10^{38}$
Plus petit $\tilde{x} \geqslant 0$ normalisé	0	0000 0001	000 0000 0000 0000 0000 0000	2^{-126}
Plus petit $\tilde{x} \geqslant 0$ dénormalisé	0	0000 0000	000 0000 0000 0000 0000 0001	2^{-149}
Infini	0	1111 1111	000 0000 0000 0000 0000 0000	`Inf`
	1	1111 1111	000 0000 0000 0000 0000 0000	`-Inf`
NaN	0	1111 1111	010 0000 0000 0000 0000 0000	`NaN`
2	0	1000 0000	000 0000 0000 0000 0000 0000	2,0
exemple 1	0	1000 0001	101 0000 0000 0000 0000 0000	6,5
exemple 2	1	1000 0001	101 0000 0000 0000 0000 0000	-6,5
exemple 3	1	1000 0101	110 1101 0100 0000 0000 0000	-118,625

Table 1.4: Exemples en simple précision tabexsimple

Détaillons l'exemple 3 : on code le nombre décimal -118,625 en utilisant le système IEEE 754.

1. C'est un nombre négatif, le bit de signe est donc "1",
2. On écrit le nombre (sans le signe) en binaire. Nous obtenons 1110110,101,
3. On décale la virgule vers la gauche, en laissant seulement un 1 sur sa gauche (nombre flottant normalisé): $1110110,101 = 1,110110101 \times 2^6$. La mantisse est la partie à droite de la virgule, remplie de 0 vers la droite pour obtenir 23 bits. Cela donne 11011010100000000000000,
4. L'exposant est égal à 6, et nous devons le convertir en binaire et le décaler. Pour le format 32-bit IEEE 754, le décalage est 127. Donc 6 + 127 = 133. En binaire, cela donne 10000101.

1.3.2 Double précision

C'est le format 64 bits : 1 bit de signe, 11 bits d'exposant (-1022 à 1023), 52 bits de mantisse. L'exposant est décalé de $2^{N_e-1}-1 = 2^{10}-1 = 1023$.

signe s	*exposant e*	*mantisse m*
$\overbrace{S}$	$\overbrace{EEEEEEEEEEE}$	$\overbrace{FFFFF.FFFFFF}$
0	1 — 11	12 — 63

Table 1.5: Représentation en double précision tab4

$\tilde{x}$	exposant e	mantisse m
$\tilde{x} = 0$ (si $S = 0$) $\tilde{x} = -0$ (si $S = 1$)	$e = 0$	$m = 0$
Nombre *normalisé* $\tilde{x} = (-1)^S \times 2^{e-1023} \times 1.m$	$0 < e < 2047$	quelconque
Nombre *dénormalisé* $\tilde{x} = (-1)^S \times 2^{e-1022} \times 0.m$	$e = 0$	$m \neq 0$
$\tilde{x} =$ `Inf` (si $S = 0$) $\tilde{x} = -$`Inf` (si $S = 1$)	$e = 2047$	$m = 0$
$\tilde{x} =$ `NaN` *(Not a Number)*	$e = 2047$	$m \neq 0$

Table 1.6: Représentation en double précision tab5

- La précision machine est $eps = 2^{-52}$.
- Le *plus grand nombre positif fini*, et le *plus petit nombre négatif fini* sont : $\tilde{x}^{\pm}_{max} = \pm(2^{1024} - 2^{971}) = \pm 1,7976931348623157 \times 10^{308}$
- **Overflow**: L'intervalle de calcul est $[\tilde{x}^{-}_{max}, \tilde{x}^{+}_{max}]$. Si un calcul produit un nombre x qui n'est pas dans cet intervalle, on dit qu'il y a *overflow*. La valeur de x est alors mise à $\pm$`Inf`. Cela peut arriver au milieu du calcul, même si le résultat final peut être représenté par un nombre machine.
- Le *plus petit nombre positif normalisé* différent de zéro, et le *plus grand nombre négatif normalisé* différent de zéro sont :
  $\tilde{x}^{\pm}_{min} = \pm 2^{-1022} = \pm 2,2250738585072020 \times 10^{-308}$
- Le système IEEE permet les calculs avec des nombres dénormalisés dans l'intervalle $[\tilde{x}^{+}_{min} * eps, \tilde{x}^{+}_{min}]$.
- **Underflow**: Si un calcul produit un nombre positif x qui est plus petit que $\tilde{x}^{+}_{min} * eps$, on dit qu'il y a *underflow*. Néanmoins, le calcul ne s'arrête pas dans ce cas, il continue avec la valeur de x mise à zéro.

1.3.3 En MATLAB

En MATLAB, les calculs réels sont en double précision par défaut. La fonction `single` peut être utilisée pour convertir les nombres en simple précision. Pour voir la représentation des nombres réels en MATLAB, on peut les afficher au format hexadécimal avec la commande *format hex*.

Le système hexadécimal est celui en base 16 et utilise 16 symboles : 0 à 9 pour représenter les valeurs de 0 à 9, et A, B, C, D, E, F pour représenter les valeurs de 10 à 15. La lecture s'effectue de droite à gauche. La valeur vaut la somme des chiffres affectés de poids correspondant aux puissances successives du nombre 16. Par exemple, $5EB52_{16}$ vaut $2 * 16^0 + 5 * 16^1 + 11 * 16^2 + 14 * 16^3 + 5 * 16^4 = 387922$. Pour passer du binaire au format hexadecimal, c'est facile en regardant la chaîne binaire en groupe de 4 chiffres, et en représentant chaque groupe en un chiffre hexadécimal. Par exemple,

$$\begin{aligned} 387922 = 01011110101101010010_2 &= 0101 \quad 1110 \quad 1011 \quad 0101 \quad 0010_2 \\ &= 5 \quad E \quad B \quad 5 \quad 2_{16} \\ &= 5EB52_{16} \end{aligned}$$

La conversion de l'hexadécimal au binaire est le processus inverse.

1.4 Calculs sur les nombres flottants

1.4.1 Erreurs d'arrondi

{secErrArrondis}

Si $\tilde{x}$ et $\tilde{y}$ sont deux nombres machine, alors $z = \tilde{x} \times \tilde{y}$ ne correspondra pas en général à un nombre machine puisque le produit demande une quantité double de chiffres. Le résultat sera un nombre machine $\tilde{z}$ proche de z.

On définit l'*erreur absolue* entre un nombre réel x et le nombre machine correspondant $\tilde{x}$ par

$$r_a = |x - \tilde{x}|.$$

L'*erreur relative* entre ces nombres (si $x \neq 0$) est définie par

$$r = \frac{|x - \tilde{x}|}{|x|}.$$

Opérations machine: On désigne par *flop* (de l'anglais *floating operation*) une opération élémentaire à virgule flottante (addition, soustraction, multiplication ou division) de l'ordinateur. Sur les calculateurs actuels on peut s'attendre à la précision suivante, obtenue dans les opérations basiques:

$$\tilde{x} \tilde{\oplus} \tilde{y} = (\tilde{x} \oplus \tilde{y})(1 + r)$$

où $|r| < eps$, la précision machine, et $\oplus$ représente l'opération exacte, $\oplus \in \{+, -, *, /\}$ et $\tilde{\oplus}$ représente l'opération de l'ordinateur (*flop*).

1.4.2 Associativité

L'associativité des opérations élémentaires comme par exemple l'addition:

$$(x + y) + z = x + (y + z),$$

n'est plus valide en arithmétique finie. Par exemple, avec 6 chiffres de précision, si on prend les trois nombres

$$x = 1.23456e - 3, \quad y = 1.00000e0, \quad z = -y,$$

on obtient $(x + y) + z = (0.00123 + 1.00000e0) - 1.00000e0 = 1.23000e - 3$ alors que $x + (y + z) = x = 1.23456e - 3$. Il est donc essentiel de considérer l'ordre des opérations et faire attention où l'on met les parenthèses.

1.4.3 Monotonicité

Supposons que l'on a une fonction f strictement croissante sur un intervalle $[a, b]$. Peut-on assurer en arithmétique finie que

$$\tilde{x} < \tilde{y} \Rightarrow f(\tilde{x}) < f(\tilde{y})?$$

En général non. Dans la norme IEEE les *fonctions standard* sont implémentées de façon à respecter la monotonicité (mais pas la stricte monotonicité).

1.4.4 Erreurs d'annulation

Ce sont les erreurs dûes à l'annulation numérique de chiffres significatifs, quand les nombres ne sont représentés qu'avec une quantité finie de chiffres, comme les nombres machine. Il est donc important en pratique d'être attentifs aux signes dans les expressions, comme l'exemple suivant le montre :

Exemple : on cherche à évaluer sur l'ordinateur de façon précise, pour de petites valeurs de x, la fonction

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

Pour $|x| < \sqrt{eps}$, on risque d'avoir le nombre $\sqrt{1 - x^2}$ remplacé par 1, par la machine, et donc lors du calcul de $f(x)$, on risque d'effectuer une division par 0. Par exemple, pour $x = \frac{\sqrt{eps}}{2}$, on obtient :

```
>> f=inline('1./(1-sqrt(1-x.^2))','x');
>> f(0.5*sqrt(eps))

ans =

   Inf
```

On ne peut donc pas évaluer précisément $f(x)$ sur l'ordinateur dans ce cas. Maintenant, si on multiplie dans $f(x)$ le numérateur et le dénominateur par $1+\sqrt{1-x^2}$, on obtient

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

Cette fois, on peut évaluer $f(x)$ de façon précise :

```
>> f=inline('(1+sqrt(1-x.^2))./x.^2','x')

>> f(0.5*sqrt(eps))

ans =

   3.6029e+16
```

C'est ce qui se passe dans la cas du calcul de π avec l'algorithme naïf 1.1. En utilisant la formule

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

comme $\sin\alpha_n \to 0$, le numérateur à droite est de la forme

$$1-\sqrt{1-\varepsilon^2}, \quad \text{avec } \varepsilon = \sin\alpha_n \text{ petit },$$

donc sujet aux erreurs d'annulation. Pour y palier, il faut reformuler les équations de façon à s'affranchir des erreurs d'annulations, par exemple en multipliant le numérateur et le dénominateur par $(1+\sqrt{1-\sin^2\alpha_n})$.

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

On peut alors écrire l'Algorithme 1.2 correspondant au calcul de π avec cette nouvelle formule.

Algorithme 1.2 Algorithme de calcul de π, version stable {AlgoPiStable}

1: $s \leftarrow 1$, $n \leftarrow 4$, ▷ Initialisations
2: **Tantque** $s > 1e-10$ **faire** ▷ Arrêt si $s = \sin(\alpha)$ est petit
3: $s \leftarrow s/sqrt(2*(1-sqrt(1-s*s)))$ ▷ nouvelle valeur de $\sin(\alpha/2)$
4: $n \leftarrow 2*n$ ▷ nouvelle valeur de n
5: $A \leftarrow (n/2)*s$ ▷ nouvelle valeur de l'aire du polygône
6: **Fin Tantque**

1.5 Quelques catastrophes dûes à l'arithmétique flottante

Il y a un petit nombre "connu" de catastrophes dans la vie réelle qui sont attribuables à une mauvaise gestion de l'arithmétique des ordinateurs (erreurs d'arondis, d'annulation), voir [3]. Dans le premier exemple ci-dessous cela c'est payé en vies humaines.

n	A_n	$\lvert A_n - \pi \rvert$	$\sin(\alpha_n)$
4	2.00000000000000	1.141593e+00	1.000000e+00
8	2.82842712474619	3.131655e-01	7.071068e-01
16	3.06146745892072	8.012519e-02	3.826834e-01
32	3.12144515225805	2.014750e-02	1.950903e-01
64	3.13654849054594	5.044163e-03	9.801714e-02
128	3.14033115695475	1.261497e-03	4.906767e-02
256	3.14127725093277	3.154027e-04	2.454123e-02
512	3.14151380114430	7.885245e-05	1.227154e-02
1024	3.14157294036709	1.971322e-05	6.135885e-03
2048	3.14158772527716	4.928313e-06	3.067957e-03
4096	3.14159142151120	1.232079e-06	1.533980e-03
8192	3.14159234557012	3.080197e-07	7.669903e-04
16384	3.14159257658487	7.700492e-08	3.834952e-04
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
262144	3.14159265328899	3.008003e-10	2.396845e-05
524288	3.14159265351459	7.519985e-11	1.198422e-05
1048576	3.14159265357099	1.879963e-11	5.992112e-06
2097152	3.14159265358509	4.699352e-12	2.996056e-06
4194304	3.14159265358862	1.174172e-12	1.498028e-06
8388608	3.14159265358950	2.926548e-13	7.490141e-07
16777216	3.14159265358972	7.238654e-14	3.745070e-07
33554432	3.14159265358978	1.731948e-14	1.872535e-07
67108864	3.14159265358979	3.552714e-15	9.362676e-08
134217728	3.14159265358979	0.000000e+00	4.681338e-08
268435456	3.14159265358979	8.881784e-16	2.340669e-08
536870912	3.14159265358979	1.332268e-15	1.170334e-08
1073741824	3.14159265358979	1.332268e-15	5.851672e-09
2147483648	3.14159265358979	1.332268e-15	2.925836e-09
4294967296	3.14159265358979	1.332268e-15	1.462918e-09
8589934592	3.14159265358979	1.332268e-15	7.314590e-10
17179869184	3.14159265358979	1.332268e-15	3.657295e-10
34359738368	3.14159265358979	1.332268e-15	1.828648e-10
68719476736	3.14159265358979	1.332268e-15	9.143238e-11

{tab6}

Table 1.7: Calcul de π avec l'algorithme stable

Missile Patriot

En février 1991, pendant la Guerre du Golfe, une batterie américaine de missiles Patriot, à Dharan (Arabie Saoudite), a échoué dans l'interception d'un missile Scud irakien. Le Scud a frappé un baraquement de l'armée américaine et a tué 28 soldats. La commission d'enquête a conclu à un calcul incorrect du temps de parcours, dû à un problème d'arrondi. Les nombres étaient représentés en virgule fixe sur 24 bits, donc 24 chiffres binaires. Le temps était compté par l'horloge interne du système en 1/10 de seconde. Malheureusement, 1/10 n'a pas d'écriture finie dans le système binaire : 1/10 = 0,1 (dans le système décimal) = 0,0001100110011001100110011... (dans le système binaire). L'ordinateur de bord arrondissait 1/10 à 24 chiffres, d'où une petite erreur dans le décompte du temps pour chaque 1/10 de seconde. Au moment de l'attaque, la batterie de missile Patriot était allumée depuis environ 100 heures, ce qui avait entraîné une accumulation des erreurs d'arrondi de 0,34 s. Pendant ce temps, un missile Scud parcourt environ 500 m, ce qui explique que le Patriot soit passé à côté de sa cible. Ce qu'il aurait fallu faire c'était redémarrer régulièrement le système de guidage du missile.

Explosion d'Ariane 5

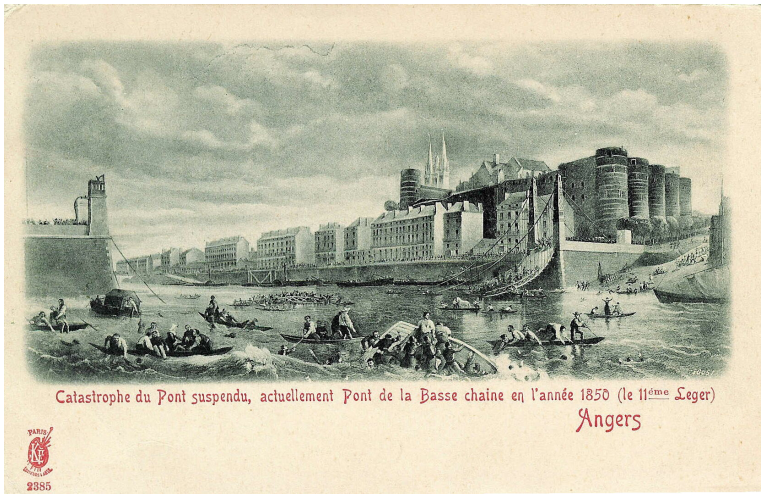
Le 4 juin 1996, une fusée Ariane 5, a son premier lancement, a explosé 40 secondes après l'allumage. La fusée et son chargement avaient coûté 500 millions de dollars. La commission d'enquête a rendu son

rapport au bout de deux semaines. Il s'agissait d'une erreur de programmation dans le système inertiel de référence. À un moment donné, un nombre codé en virgule flottante sur 64 bits (qui représentait la vitesse horizontale de la fusée par rapport à la plate-forme de tir) était converti en un entier sur 16 bits. Malheureusement, le nombre en question était plus grand que 32768 (overflow), le plus grand entier que l'on peut coder sur 16 bits, et la conversion a été incorrecte.

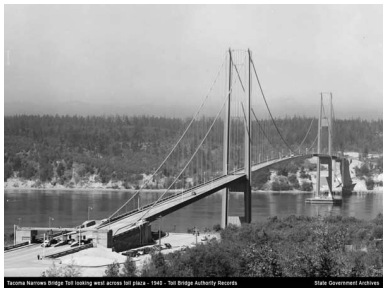
Bourse de Vancouver

Un autre exemple où les erreurs de calcul on conduit à une erreur notable est le cas de l'indice de la Bourse de Vancouver. En 1982, elle a crée un nouvel indice avec une valeur nominale de 1000. Après chaque transaction boursière, cet indice était recalculé et tronqué après le troisième chiffre décimal et, au bout de 22 mois, la valeur obtenue était 524,881, alors que la valeur correcte était 1098.811. Cette différence s'explique par le fait que toutes les erreurs d'arrondi étaient dans le même sens : l'opération de troncature diminuait à chaque fois la valeur de l'indice

1.6 Essai



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



(b) Takoma Narrows Bridge, Washington (1940)

(c) Millennium Bridge, London (2000)

Figure 1.4: Une histoire de ponts

Chapitre 2

Langage algorithmique

2.1 Pseudo-langage algorithmique

Pour uniformiser l'écriture des algorithmes nous employons, un pseudo-langage contenant l'indispensable :

- variables,
- opérateurs (arithmétiques, relationnels, logiques),
- expressions,
- instructions (simples et composées),
- fonctions.

Ce pseudo-langage sera de fait très proche du langage de programmation de Matlab.

2.1.1 Données et constantes

Une donnée est une valeur introduite par l'utilisateur (par ex. une température, une vitesse, ... Une constante est un symbole ou un identificateur non modifiable (par ex. π, la constante de gravitation,...)

2.1.2 Variables

Definition 2.1

Une variable est un objet dont la valeur est modifiable, qui possède un nom et un type (entier, charactère, réel, complexe, ...). Elle est rangée en mémoire à partir d'une certaine adresse.

2.1.3 Opérateurs

Opérateurs arithmétiques

Nom	Symbole	example
addition	$+$	$a+b$
soustraction	$-$	$a-b$
opposé	$-$	$-a$
produit	$*$	$a*b$
division	$/$	a/b
puissance a^b	^	$a\hat{}b$

Table 2.1: Opérateurs arithmétiques

Opérateurs relationnels

Nom	Symbole	example	Commentaires
identique	$==$	$a == b$	vrai si a et b ont même valeur, faux sinon.
différent	$\sim=$	$a \sim= b$	faux si a et b ont même valeur, vrai sinon.
inférieur	$<$	$a < b$	vrai si a est plus petit que b, faux sinon.
supérieur	$>$	$a > b$	vrai si a est plus grand que b, faux sinon.
inférieur ou égal	$<=$	$a <= b$	vrai si a est plus petit ou égal à b, faux sinon.
supérieur ou égal	$>=$	$a >= b$	vrai si a est plus grand ou égal à b, faux sinon.

Table 2.2: Opérateurs relationnels

Opérateurs logiques

Nom	Symbole	example	Commentaires
négation	$\sim$	$\sim a$	vrai si a est faux (ou nul), faux sinon.
ou	$\vert$	$a\vert b$	vrai si a **ou** b est vrai (non nul), faux sinon.
et	$\&$	$a\&b$	vrai si a **et** b sont vrais (non nul), faux sinon.

Table 2.3: Opérateurs logiques

Opérateur d'affectation

Nom	Symbole	example	Commentaires
affectation	$\leftarrow$	$a \leftarrow b$	On affecte à la variable a le contenu de b

Table 2.4: Opérateurs d'affectation

2.1.4 Expressions

Definition 2.2

Une expression est un 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 2.3 • Voici un exemple classique d'expression numérique :

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

On appelle **opérandes** les identifiants a, b et c, et les nombres 4 et 2. Les symboles $*$, $-$ et $/$ sont les **opérateurs**.

- Voici un exemple classique d'expression booléenne (logique) :

$$(x < 3.14)$$

Dans cette expression, x est une variable numérique et 3.14 est un nombre réel. Cette expression prendra la valeur vrai (i.e. 1) si x est plus grand que 3.14. Sinon, elle prendra la valeur faux (i.e. 0)

2.1.5 Instructions

Definition 2.4

Une **instruction** est un ordre ou un groupe d'ordres qui déclenche l'exécution de certaines actions par l'ordinateur. Il y a deux types d'instructions : simple et structuré.
Les **instructions simples** sont essentiellement des ordres seuls et inconditionnels réalisant l'une des tâches suivantes :

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

Les **instructions structurées** sont essentiellement :

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

Les exemples qui suivent sont écrits dans un pseudo langage algorithmique mais sont facilement transposable dans la plupart des langages de programmation.

Instructions simples

Voici un exemple de l'*instruction simple d'affectation* :

1: $a \leftarrow 3.14 * R$

On évalue l'expression $3.14 * R$ et affecte le résultat à la variable a.
Un autre exemple est donné par l'*instruction simple d'affichage* :

```
affiche('bonjour')
```

Affiche la chaine de caractères `'bonjour'` à l'écran. Cette instruction fait appel à la fonction `affiche`.

Instructions composées

Instructions répétitives, boucle «pour»

Algorithme 2.1 Exemple de boucle «pour»

Données : n : un entier.

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

Instruction répétitive, boucle «tant que»

Algorithme 2.2 Exemple de boucle «tant que»

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

Instruction répétitive, boucle «répéter ...jusqu'à»

Algorithme 2.3 Exemple de boucle «répéter ...jusqu'à»

```
1: i ← 0, x ← 1
2: Répéter
3:     x ← x + i * i
4:     i ← i + 1
5: jusqu'à i>=1000
```

Instructions conditionnelles «si»

Algorithme 2.4 Exemple d'instructions conditionnelle «si»

Données : *age* : un réel.

```
1: Si age >= 18 alors
2:     affiche('majeur')
3: Sinon Si age >= 0 alors
4:     affiche('mineur')
5: Sinon
6:     affiche('en devenir')
7: Fin Si
```

2.1.6 Fonctions

Les fonctions permettent

- d'automatiser certaines tâches répétitives au sein d'un même programme,
- d'ajouter à la clarté d'un programme,
- l'utilisation de portion de code dans un autre programme,
- ...

Fonctions prédéfinies

Pour faciliter leur usage, tous les langages de programmation possèdent des fonctions prédéfinies. On pourra donc supposer que dans notre langage algorithmique un grand nombre de fonctions soient prédéfinies : par exemple, les fonctions mathématiques sin, cos, exp, abs, $\cdots$ (pour ne citer quelles)

Syntaxe

On utilise la syntaxe suivante pour la définition d'une fonction

Fonction $[args_1, \ldots, args_n] \leftarrow$ NomFonction($arge_1, \ldots, arge_m$)
instructions
Fin Fonction

La fonction se nomme NomFonction. Elle admet comme paramètres d'entrée (données) les m arguments $arge_1, \ldots, arge_m$ et comme paramètres de sortie (résultats) les n arguments $args_1, \ldots, args_n$. Ces derniers doivent être déterminés dans le corps de la fonction (partie instructions).
Dans le cas ou la fonction n'admet qu'un seul paramètre de sortie, l'écriture se simplifie :

Fonction $args \leftarrow$ NomFonction($arge_1, \ldots, arge_m$)
instructions
Fin Fonction

Ecrire ses propres fonctions

Pour écrire une fonction «propre», il faut tout d'abord déterminer exactement ce que devra faire cette fonction.

Puis, il faut pouvoir répondre à quelques questions :

1. Quelles sont les données (avec leurs limitations)?
2. Que doit-on calculer ?

Et, ensuite il faut la **commenter** : expliquer son usage, type des paramètres, ...

Exemple : résolution d'une équation du premier degré

Nous voulons écrire une fonction calculant la solution de l'équation

$$ax + b = 0,$$

où nous supposons que $a \in \mathbb{R}^*$ et $b \in \mathbb{R}$. La solution de ce problème est donc

$$x = -\frac{b}{a}.$$

Les données de cette fonction sont $a \in \mathbb{R}^*$ et $b \in \mathbb{R}$. Elle doit retourner $x = -\frac{b}{a}$ solution de $ax + b = 0$.

Algorithme 2.5 Exemple de fonction : Résolution de l'équation du premier degré $ax + b = 0$.

Données : a : nombre réel différent de 0
b : nombre réel.
Résultat : x : un réel.

1: **Fonction** $x \leftarrow$ REPD(a, b)
2: $x \leftarrow -b/a$
3: **Fin Fonction**

Remarque 2.5 Cette fonction est très simple, toutefois pour ne pas «alourdir» le code nous n'avons pas vérifié la validité des données fournies.

Exercice 2.1.1

Ecrire un algorithme permettant de valider cette fonction.

Exemple : résolution d'une équation du second degré

Nous cherchons les solutions réelles de l'équation

$$ax^2 + bx + c = 0, \tag{2.1}$$ {eq-snd-de

où nous supposons que $a \in \mathbb{R}^*$, $b \in \mathbb{R}$ et $c \in \mathbb{R}$ sont donnés.

Mathématiquement, l'étude des solutions réelles de cette équation nous amène à envisager trois cas suivant les valeurs du discriminant $\Delta = b^2 - 4ac$

- si $\Delta < 0$ alors les deux solutions sont complexes,
- si $\Delta = 0$ alors la solution est $x = -\frac{b}{2a}$,
- si $\Delta > 0$ alors les deux solutions sont $x_1 = \frac{-b-\sqrt{\Delta}}{2*a}$ et $x_2 = \frac{-b-\sqrt{\Delta}}{2*a}$.

Exercice 2.1.2

1. Ecrire la fonction `discriminant` permettant de calculer le discriminant de l'équation (2.1).
2. Ecrire la fonction `RESD` permettant de résoudre l'équation (2.1) en utilisant la fonction `discriminant`.
3. Ecrire un programme permettant de valider ces deux fonctions.

Exercice 2.1.3

Même question que précédemment dans le cas complexe (solution et coefficients).

2.2 Méthodologie d'élaboration d'un algorithme

2.2.1 Description du problème

- Spécification d'un ensemble de données
  Origine : énoncé,hypothèses, sources externes, ...
- Spécification d'un ensemble de buts à atteindre
  Origine : résultats, opérations à effectuer, ...
- Spécification des contraintes

2.2.2 Recherche d'une méthode de résolution

- Clarifier l'énoncé.
- Simplifier le problème.
- Ne pas chercher à le traiter directement dans sa globalité.
- S'assurer que le problème est soluble (sinon problème d'indécidabilité!)
- Recherche d'une stratégie de construction de l'algorithme
- Décomposer le problème en sous problèmes partiels plus simples : raffinement.
- Effectuer des raffinements successifs.
- Le niveau de raffinement le plus élémentaire est celui des instructions.

2.2.3 Réalisation d'un algorithme

Il doit être conçu indépendamment du langage de programmation et du système informatique (sauf cas très particulier)

- L'algorithme doit être exécuté en un nombre fini d'opérations.
- L'algorithme doit être spécifié clairement, sans la moindre ambiguïté.
- Le type de données doit être précisé.
- L'algorithme doit fournir au moins un résultat.
- L'algorithme doit être effectif : toutes les opérations doivent pouvoir être simulées par un homme en temps fini.

Pour écrire un algorithme détaillé, il faut tout d'abord savoir répondre a quelques questions :

- Que doit-il faire ? (i.e. Quel problème est-il censé résoudre?)
- Quelles sont les données necessaires à la résolution de ce problème?
- Comment résoudre ce problème «à la main» (sur papier)?

Si l'on ne sait pas répondre à l'une de ces questions, l'écriture de l'algorithme est fortement comprise.

2.2.4 Exercices

Exercice 2.2.1: Algorithme pour une somme {exo:algo:02}

Ecrire un algorithme permettant de calculer

$$S(x) = \sum_{k=1}^{n} k \sin(2 * k * x)$$

Correction Exercice 2.2.1 L'énoncé de cet exercice est imprécis. On choisi alors $x \in \mathbb{R}$ et $n \in \mathbb{N}$ pour rendre possible le calcul. Le problème est donc de calculer

$$\sum_{k=1}^{n} k \sin(2kx).$$

Toutefois, on aurait pu choisir $x \in \mathbb{C}$ ou encore un tout autre problème :

$$\text{Trouver } x \in \mathbb{R} \text{ tel que } S(x) = \sum_{k=1}^{n} k \sin(2kx)$$

où $n \in \mathbb{N}$ et S, fonction de $\mathbb{R}$ à valeurs réelles, sont les données!

Algorithme 2.6 Calcul de $S = \sum_{k=1}^{n} k \sin(2kx)$

Données : x : nombre réel,
n : nombre entier.
Résultat : S : un réel.

1: $S \leftarrow 0$
2: **Pour** $k \leftarrow 1$ **à** n **faire**
3: $S \leftarrow S + k * \sin(2 * k * x)$
4: **Fin Pour**

◇

Exercice 2.2.2: Algorithme pour un produit

{exo:algo:03}

Ecrire un algorithme permettant de calculer

$$P(z) = \prod_{n=1}^{k} sin(2 * k * z/n)^k$$

Correction Exercice 2.2.2 L'énoncé de cet exercice est imprécis. On choisi alors $z \in \mathbb{R}$ et $k \in \mathbb{N}$ pour rendre possible le calcul.

Algorithme 2.7 Calcul de $P = \prod_{n=1}^{k} \sin(2kz/n)^k$

Données : z : nombre réel,
k : nombre entier.
Résultat : P : un réel.

1: $P \leftarrow 1$
2: **Pour** $n \leftarrow 1$ **à** k **faire**
3: $P \leftarrow P * \sin(2 * k * z/n)\hat{}k$
4: **Fin Pour**

◇

Exercice 2.2.3: Série de Fourier

{exo:algo:06}

Soit la série de Fourier

$$x(t) = \frac{4A}{\pi}\left\{\cos\omega t - \frac{1}{3}\cos 3\omega t + \frac{1}{5}\cos 5\omega t - \frac{1}{7}\cos 7\omega t + \cdots\right\}.$$

Ecrire la fonction SFT permettant de calculer $x_n(t)$.

Correction Exercice 2.2.3 Nous devons écrire la fonction permettant de calculer

$$x_n(t) = \frac{4A}{\pi}\sum_{k=1}^{n}(-1)^{k+1}\frac{1}{2k-1}\cos((2k-1)\omega t)$$

Les données de la fonction sont $A \in \mathbb{R}$, $\omega \in \mathbb{R}$, $n \in \mathbb{N}^*$ et $t \in \mathbb{R}$.

Grâce a ces renseignements nous pouvons déjà écrire l'entête de la fonction :

Algorithme 2.8 En-tête de la fonction SFT retournant valeur de la série de Fourier en t tronquée au n premiers termes de l'exercice 2.2.3.

Données : t : nombre réel,
n : nombre entier strictement positif
A, ω : deux nombres réels.
Résultat : x : un réel.

1: **Fonction** $x \leftarrow$ SFT(t, n, A, ω)
2: ...
3: **Fin Fonction**

Maintenant nous pouvons écrire progressivement l'algorithme pour aboutir au final à une version ne contenant que des opérations élémentaires.

Algorithme 2.9 $\mathcal{R}_0$

1: $x \leftarrow \frac{4A}{\pi} \sum_{k=1}^{n} \left(\begin{array}{c} (-1)^{k+1} \frac{1}{2k-1} \times \\ \cos((2k-1)\omega t) \end{array} \right)$

Algorithme 2.9 $\mathcal{R}_1$

1: $S \leftarrow \sum_{k=1}^{n} (-1)^{k+1} \frac{1}{2k-1} \cos((2k-1)\omega t)$
2: $x \leftarrow \frac{4A}{\pi} S$

Algorithme 2.9 $\mathcal{R}_2$

1: $S \leftarrow \sum_{k=1}^{n} (-1)^{k+1} \frac{1}{2k-1} \cos((2k-1)\omega t)$
2: $x_n(t) \leftarrow \frac{4A}{\pi} S$

Algorithme 2.9 $\mathcal{R}_3$

1: $S \leftarrow 0$
2: **Pour** $k = 1$ **à** n **faire**
3: $\quad S \leftarrow S + (-1)^{k+1} \frac{1}{2k-1} * \cos((2k-1)\omega t)$
4: **Fin Pour**
5: $x_n(t) \leftarrow \frac{4A}{\pi} S$

Finalement la fonction est

Algorithme 2.9 Fonction SFT retournant la valeur de la série de Fourier en t tronquée au n premiers termes de l'exercice ??.

Données : t : nombre réel,
n : nombre entier strictement positif
$A,\ \omega$: deux nombres réels.
Résultat : x : un réel.

```
1: Fonction x ← SFT( t, n, A, ω )
2:     S ← 0
3:     Pour k = 1 à n faire
4:         S ← S + ((−1)^(k + 1)) ∗ cos((2 ∗ k − 1) ∗ ω ∗ t)/(2 ∗ k − 1)
5:     Fin Pour
6:     S ← 4 ∗ A ∗ S/π
7: Fin Fonction
```

◇

Exercice 2.2.4

Reprendre les trois exercices précédents en utilisant les boucles «tant que».

2.3 Principes de «bonne» programmation pour attaquer de «gros» problèmes

Tous les exemples vus sont assez courts. Cependant, il peut arriver que l'on ait des programmes plus longs à écrire (milliers de lignes, voir des dizaines de milliers de lignes). Dans l'industrie, il arrive que des équipes produisent des codes de millions de lignes, dont certains mettent en jeux des vies humaines (contrôler un avion de ligne, une centrale nucléaire, ...). Le problème est évidemment d'écrire des programmes sûrs. Or, *un programme à 100% sûr, cela n'existe pas!* Cependant, plus un programme est simple, moins le risque d'erreur est grand : c'est sur cette remarque de bon sens que se basent les «bonnes» méthodes. Ainsi :

Tout problème compliqué doit être découpé en sous-problèmes plus simples

Il s'agit, lorsqu'on a un problème P à résoudre, de l'analyser et de le décomposer en un ensemble de problèmes P_1, P_2, $P_3, \ldots$ plus simples. Puis, P_1, est lui-même analysé et décomposé en $P_{11}, P_{12}, \ldots$, et

P_2 en P_{21}, P_{22}, etc. On poursuit cette analyse jusqu'à ce qu'on n'ait plus que des problèmes élémentaires à résoudre. Chacun de ces problèmes élémentaires est donc traité séparément dans un module, c'est à dire un morceau de programme relativement indépendant du reste. Chaque module sera *testé et validé* séparément dans la mesure du possible et naturellement *largement documenté*. Enfin ces modules élémentaires sont assemblés en modules de plus en plus complexes, jusqu'à remonter au problème initiale. A chaque niveau, il sera important de bien réaliser les phases de test, validation et documentation des modules.

Par la suite, on s'évertue à écrire des algorithmes!
Ceux-ci ne seront pas optimisés[a]!

[a] améliorés pour minimiser le nombre d'opérations élémentaires, l'occupation mémoire, ..

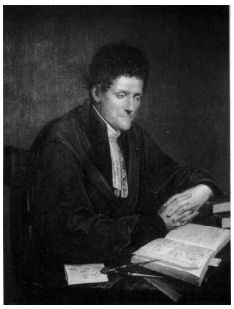
Chapitre 3

Résolution de systèmes non linéaires

Un problème *simple* comme la recherche des zéros/racines d'un polynôme n'est pas ... *simple*. Depuis tout petit, on sait trouver les racines d'un polynôme de degré 2 : $a_2x^2 + a_1x + a_0 = 0$. Quid des racines de polynômes de degré plus élevé?

- **degré 2** : Babylonniens en 1600 avant J.-C.
- **degré 3** : *Scipio del Ferro* (1465-1526, mathématicien italien) et *Niccolo Fontana* (1499-1557, mathématicien italien)
- **degré 4** : *Ludovico Ferrari* (1522-1565, mathématicien italien)
- **degré 5** : *Paolo Ruffini* (1765-1822, mathématicien italien) en 1799, *Niels Henrick Abel* (1802-1829, mathématicien norvégien) en 1824, montrent qu'il n'existe **pas de solution analytique**.

(a) *Niccolo Fontana* 1499-1557, mathématicien italien



(b) *Paolo Ruffini* 1765-1822, mathématicien italien

(c) *Niels Henrick Abel* 1802-1829, mathématicien norvégien

L'objectif de ce chapitre est de rechercher **numériquement** les *zéros/racines* d'une fonction ou d'un systéme d'équations lorsqu'ils existent :

- Soit $f : [a, b] \subset \mathbb{R} \longrightarrow \mathbb{R}$, trouver $x \in [a, b]$ tel que $f(x) = 0$, étudier en section 3.1
- Soit $\boldsymbol{f} : \Omega \subset \mathbb{K}^n \longrightarrow \mathbb{K}^n$, trouver $\boldsymbol{x} \in \Omega$ tel que $\boldsymbol{f}(\boldsymbol{x}) = 0$ ($\mathbb{K} = \mathbb{R}$ ou $\mathbb{K} = \mathbb{C}$). étudier en section 3.3

3.1 Recherche des zéros d'une fonction

{RSNL:sec

Soit $f : [a, b] \subset \mathbb{R} \longrightarrow \mathbb{R}$ une fonction continue. On cherche à déterminer les zéros de f ou plus précisément l'ensemble des $x \in [a, b]$ tels que $f(x) = 0$.

Dans un premier temps, on étudiera la **méthode de dichotomie** qui est *assez naturelle*. Puis on étudiera plusieurs algorithmes liés à la méthode du point fixe.

3.1.1 Méthode de dichotomie ou de bissection

Principe et résultats

principe de la méthode de dichotomie : Soit I un intervalle contenant un unique zéro de la fonction f, on le divise par son milieu en deux intervalles et on détermine lequel des deux contient le zéro. On itére ce processus sur le nouvel intervalle.

Plus précisement, on suppose que la fonction f vérifie $f(a)f(b) < 0$ et qu'il existe un unique $\xi \in]a, b[$ tel que $f(x) = 0$.

On note $I^{(0)} =]a, b[$ et $x_0 = (a+b)/2$. Si $f(x_0) = 0$ alors on a fini! Supposons $f(x_0) \neq 0$, alors ξ appartient à $]a, x_0[$ si $f(a)f(x_0) < 0$, sinon il appartient à $]x_0, b[$. On vient donc de déterminer une méthode permettant de diviser par 2 la longueur de l'intervalle de recherche de ξ. On peut bien évidemment itérer ce principe en définissant les trois suites $(a_k)_{k\in\mathbb{N}}$, $(b_k)_{k\in\mathbb{N}}$ et $(x_k)_{k\in\mathbb{N}}$ par

- $a_0 = a$, $b_0 = b$ et $x_0 = \frac{a+b}{2}$,
- $\forall k \in \mathbb{N}$, $x_k = (a_k + b_k)/2$, et

$$\begin{cases} a_{k+1} = b_{k+1} = x_k & \text{si } f(x_k) = 0, \\ a_{k+1} = x_k,\ b_{k+1} = b_k & \text{si } f(b_k)f(x_k) < 0, \\ a_{k+1} = a_k,\ b_{k+1} = x_k & \text{sinon (i.e. } f(a_k)f(x_k) < 0.) \end{cases}$$

En Figure 3.2, on représente les intervalles successifs obtenus par la méthode de dichotomie pour $a = 0.1$, $b = 2.4$ et $f : x \mapsto (x + 2)(x + 1)(x - 1)$.

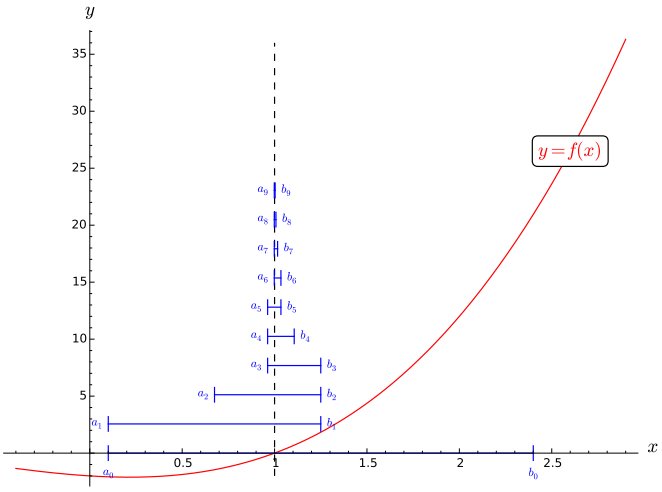


Figure 3.2: Méthode de dichotomie: $f(x) = (x + 2)(x + 1)(x - 1)$, fig:dichotomie:01

Exercice 3.1.1

[RSNL:exo102]

On suppose que la fonction f est continue sur $[a,b]$, vérifie $f(a)f(b)<0$ et qu'il existe un unique $\xi \in]a,b[$ tel que $f(x)=0$.

Q. 1 *1. Montrer que les suites (a_k) et (b_k) convergent vers α.*

2. En déduire que la suite (x_k) converge vers α.

Q. 2 *1. Montrer que pour tout $k\in\mathbb{N}$, $|x_k-\alpha|\leqslant \frac{b-a}{2^{k+1}}$.*

2. Soit $\epsilon>0$. En déduire que si $k\geqslant \frac{\log(\frac{b-a}{\epsilon})}{\log(2)}-1$ alors $|x_k-\alpha|\leqslant\epsilon$.

Correction Exercice 3.1.1

Q. 1 1. Supposons qu'il existe $k\in\mathbb{N}$ tel que $f(x_k)=0$, (i.e. $x_k=\alpha$ car $x_k\in[a,b]$) alors par construction $a_{k+i}=b_{k+i}=x_{k+i}=\alpha$ pour tout $i\in\mathbb{N}^*$. Ceci assure la convergence des 3 suites vers α.

Supposons maintenant que $\forall k\in\mathbb{N}$, $f(x_k)\neq 0$. Par construction, nous avons $a_k\leqslant a_{k+1}\leqslant b$, $a\leqslant b_{k+1}\leqslant b_k$ et $a_k\leqslant b_k$. La suite (a_k) est convergente car elle est croissante et majorée. La suite (b_k) est décroissante et minorée : elle est donc convergente. De plus $0\leqslant b_k-a_k\leqslant \frac{b_{k-1}-a_{k-1}}{2}$ et donc $0\leqslant b_k-a_k\leqslant\frac{b-a}{2^{k+1}}$. On en déduit que les suites (a_k) et (b_k) ont même limite. Comme par construction, $\forall k\in\mathbb{N}$, $\alpha\in[a_k,b_k]$ ceci entraine que α est la limite de ces 2 suites.

2. Par construction, $\forall k\in\mathbb{N},$, $a_k\leqslant x_k\leqslant b_k$ et comme les suites (a_k) et (b_k) convergent vers α, la suite (x_k) converge aussi vers α.

Q. 2 1. On a $\forall k\in\mathbb{N}$, $a_k\leqslant x_k\leqslant b_k$ et $a_k\leqslant\alpha\leqslant b_k$ d'où $|x_k-\alpha|\leqslant b_k-a_k$. Ce qui donne

$$|x_k-\alpha|\leqslant b_k-a_k\leqslant\frac{b_{k-1}-a_{k-1}}{2}\leqslant\frac{b-a}{2^{k+1}}.$$

2. Pour avoir $|x_k-\alpha|\leqslant\epsilon$, il suffit d'avoir $\frac{b-a}{2^{k+1}}\leqslant\epsilon$, et donc $k\geqslant\frac{\log(\frac{b-a}{\epsilon})}{\log(2)}-1$.

◇

On peut alors écrire la proposition suivante :

Proposition 3.1 {prop:dichotomie}

Soit $f:[a,b]\subset\mathbb{R}\longrightarrow\mathbb{R}$ une fonction continue vérifiant $f(a)f(b)<0$ et admettant $\alpha\in]a,b[$ comme **unique** solution de $f(x)=0$. Alors la suite $(x_k)_{k\in\mathbb{N}}$ définie par la méthode de dichotomie converge vers α et

$$|x_k-\alpha|\leqslant\frac{b-a}{2^{k+1}},\quad\forall k\in\mathbb{N}.$$

On a alors $\forall\epsilon>0$, $\forall k\geqslant\frac{\log(\frac{b-a}{\epsilon})}{\log(2)}-1$

$$|x_k-\alpha|\leqslant\epsilon.$$

A partir de ce résultat, on va proposer plusieurs variantes de l'algorithme de dichotomie.

On note tout d'abord que pour déterminer α il faut calculer la limite d'une suite et donc une infinité de termes! Numériquement et algorithmiquement, on se limite donc à déterminer une approximation de α. La proposition 3.1 permet d'obtenir une approximation α_ϵ de α en un nombre fini d'itérations avec une précision ϵ donnée: on choisi $\alpha_\epsilon=x_{k_{\min}}$ avec $k_{\min}=E(\frac{\log(\frac{b-a}{\epsilon})}{\log(2)})$ où $E(.)$ est la fonction *partie entière*.

Algorithmique

Tout d'abord nous allons poser "correctement" le problème pour lever toute ambiguité. En effet, si le problème est *rechercher une racine de f sur l'intervalle $[a,b]$ par la méthode de dichotomie*, dois-je uniquement rechercher une approximation d'une racine ou calculer la suite (x_k) ou l'ensemble des suites (x_k), (a_k) et (b_k)? C'est ce que nous appelerons le(s) **résultat**(s)/**objectif**(s) de l'algorithme. L'écriture

d'un algorithme est fortement corrélée aux objectifs souhaités.
Sauf note contraire, on choisira dans la suite comme objectif de déterminer une approximation de la racine α de f :

Résultat : α_ϵ : un réel tel que $|\alpha_\epsilon - \alpha| \leqslant \epsilon$.

Ensuite, pour aboutir à cet objectif on détermine les données (avec hypothèses) nécessaires et suffisantes :

Données : $a,\ b$: deux réels $a < b$,
f : $f : [a,b] \subset \mathbb{R} \longrightarrow \mathbb{R}$ vérifiant les hypothèses de la proposition 3.1,
ϵ : un réel strictement positif.

On peut alors noter (formellement pour le moment) DICHOTOMIE la fonction permettant, à partir des données f, a, b, et ϵ, de trouver α_ϵ. Cette fonction algorithmique aura la syntaxe suivante :

$$\alpha_\epsilon \leftarrow \text{DICHOTOMIE}(f, a, b, \epsilon)$$

Nous allons maintenant proposer une manière d'aborder l'écriture de cette fonction dans un langage algorithmique présenté dans []. On doit s'affranchir de tout (ou presque) symbolismes mathématiques pour s'approcher au plus près des langages de programmation. Par exemple, la notation mathématique $(x_k)_{k=0}^{10}$ pour noter les 11 premiers itérés d'une suite n'est pas directement disponible dans les langages de programmations : on peut par exemple passer par un tableau `X` de dimension 11 (au moins) pour stocker les valeurs de la suite. Suivant les langages, l'accès aux composantes d'un tableau diffère : sous Matlab/Octave l'accès au 1er élément d'un tableau `X` se fait par `X(1)` et en C/C++/Python par `X[0]`. Lors de l'écriture d'un algorithme, nous utiliserons l'opérateur `()` ou(exclusif) `[]` pour accéder aux différentes composantes d'un tableau : `()` si le 1er élément est d'indice 1 ou `[]` si le 1er élément est d'indice 0.
Pour écrire un algorithme non trivial, nous utiliserons une *technique de raffinements d'algorithme* (voir []) : par raffinements succéssifs, nous allons écrire des algorithmes **équivalents** pour aboutir au final à un algorithme n'utilisant que

- des instructions élémentaires (affectation, addition, somme, ...)
- des instructions composées (conditionnelle, boucles pour, tantque, ...)
- des fonctions usuelles, mathématiques (cos, exp, E partie entière, ...), entrées/sorties , ...

présentent dans tous les langages.
L'idée est donc de partir d'un algorithme formel facile à comprendre puis de le détailler de plus en plus au fur et à mesure des raffinements. Le passage entre deux raffinements successifs doit être simple.
Un petit rappel pour les algorithmes qui suivent : les données sont supposées ... données!

Algorithme 3.1 $\mathcal{R}_0$

1: $k_{\min} \leftarrow \text{E}(\frac{\log(\frac{b-a}{\epsilon})}{\log(2)})$ ▷ E, partie entière
2: Calcul de la suite $(x_k)_{k=0}^{k_{\min}}$ par dichotomie
3: $\alpha_\epsilon \leftarrow x_{k_{\min}}$

Algorithme 3.1 $\mathcal{R}_1$

1: $k_{\min} \leftarrow \text{E}(\frac{\log(\frac{b-a}{\epsilon})}{\log(2)})$ ▷ E, partie entière
2: Initialisation de x_0
3: **Pour** $k \leftarrow 0$ **à** $k_{\min} - 1$ **faire**
4: Calcul de la suite (x_{k+1}) par dichotomie
5: **Fin Pour**
6: $\alpha_\epsilon \leftarrow x_{k_{\min}}$

Entre les raffinements $\mathcal{R}_0$ et $\mathcal{R}_1$, nous avons juste décrit le principe de calcul de toute suite récurrente d'ordre 1.
Pour faciliter la lecture, nous rappelons la méthode de dichotomie :

- $a_0 = a$, $b_0 = b$ et $x_0 = \frac{a+b}{2}$,
- $\forall k \in [\![0, k_{\min} - 1]\!]$,

$$\begin{cases} a_{k+1} = b_{k+1} = x_k & \text{si } f(x_k) = 0, \\ a_{k+1} = x_k,\ b_{k+1} = b_k & \text{si } f(b_k)f(x_k) < 0, \\ a_{k+1} = a_k,\ b_{k+1} = x_k & \text{sinon (i.e. } f(a_k)f(x_k) < 0.) \end{cases}$$

et

$$x_{k+1} = \frac{a_{k+1} + b_{k+1}}{2}$$

Ecrit sous cette forme, le calcul de la suite (x_k) nécessite le calcul simultané des suites (a_k) et (b_k). 1

Algorithme 3.1 $\mathcal{R}_1$

1: $k_{\min} \leftarrow \mathrm{E}(\frac{\log(\frac{b-a}{\epsilon})}{\log(2)})$ ▷ E, partie entière
2: Initialisation de x_0
3: **Pour** $k \leftarrow 0$ **à** $k_{\min} - 1$ **faire**
4: Calcul de la suite (x_{k+1}) par dichotomie
5: **Fin Pour**
6: $\alpha_\epsilon \leftarrow x_{k_{\min}}$

Algorithme 3.1 $\mathcal{R}_2$

1: $k_{\min} \leftarrow \mathrm{E}(\frac{\log(\frac{b-a}{\epsilon})}{\log(2)})$
2: $a_0 \leftarrow a,\ b_0 \leftarrow b$
3: $x_0 \leftarrow \frac{a_0+b_0}{2}$
4: **Pour** $k \leftarrow 0$ **à** $k_{\min} - 1$ **faire** ▷ Calcul de x_{k+1}
5: **Si** $f(x_k) == 0$ **alors**
6: $a_{k+1} \leftarrow x_k,\ b_{k+1} \leftarrow x_k$
7: **Sinon Si** $f(x_k)f(b_k) < 0)$ **alors**
8: $a_{k+1} \leftarrow x_k,\ b_{k+1} \leftarrow b_k$
9: **Sinon**
10: $a_{k+1} \leftarrow a_k,\ b_{k+1} \leftarrow x_k$
11: **Fin Si**
12: $x_{k+1} \leftarrow \frac{a_{k+1}+b_{k+1}}{2}$
13: **Fin Pour**
14: $\alpha_\epsilon \leftarrow x_{k_{\min}}$

2
3

La transcription de ce raffinement dans un langage de programmation n'est pas forcément triviale pour tout le monde. Quid de a_k, ϵ, ... qui sont syntaxiquement incorrectes dans un langage de programmation? Le cas du ϵ est très simple : les lettres grecques étant prohibées, on la remplacera par exemple par `eps`. Pour les suites (a_k), (b_k) et (x_k), tronquées à $k_{\min}$, on pourra utiliser des tableaux (vecteurs) de $k_{\min} + 1$ réels notés $\boldsymbol{A}$, $\boldsymbol{B}$ et $\boldsymbol{X}$ qui contiendront respectivement l'ensemble des valeurs a_k, b_k et x_k pour $0 \leqslant k \leqslant k_{\min}$. Plus précisément nous pourrons utiliser les opérateurs () (indice des tableaux commence à 1) ou [] (indice des tableaux commence à 0) pour accéder aux différents éléments des tableaux et nous aurons par convention $\boldsymbol{A}(k+1) = \boldsymbol{A}[k] = a_k,\ \forall k \in [\![0, k_{\min}]\!]$. Par la suite nous utiliserons les opérateurs () et nous aurons alors

$$\forall k \in [\![0, k_{\min}]\!],\ \boldsymbol{A}(k+1) = a_k,\ \boldsymbol{B}(k+1) = b_k \text{ et } \boldsymbol{X}(k+1) = x_k.$$

En utilisant ces changements de notations nous obtenons (enfin) l'Algorithme 3.1. 4

Algorithme 3.1 Méthode de dichotomie : version 1 {Algo:dich

Données : $a,\ b$: deux réels $a < b$,
f : $f : [a,b] \subset \mathbb{R} \longrightarrow \mathbb{R}$ vérifiant les hypothèses de la proposition 3.1,
`eps` : un réel strictement positif.
Résultat : x : un réel tel que $|x - \alpha| \leqslant$ `eps`.

1: **Fonction** $x \leftarrow$ Dichotomie1($f, a, b,$ `eps`)
2: `kmin` $\leftarrow$ E($\log((b-a)/$`eps`$)/\log(2)$)
3: $\boldsymbol{A},\ \boldsymbol{B},\ \boldsymbol{X} \in \mathbb{R}^{\texttt{kmin}+1}$ ▷ $\boldsymbol{A}(k+1)$ contiendra a_k, ...
4: $\boldsymbol{A}(1) \leftarrow a,\ \boldsymbol{B}(1) \leftarrow b,\ \boldsymbol{X}(1) \leftarrow (a+b)/2$
5: **Pour** $k \leftarrow 1$ **à** `kmin` **faire**
6: **Si** $f(\boldsymbol{X}(k)) == 0$ **alors**
7: $\boldsymbol{A}(k+1) \leftarrow \boldsymbol{X}(k),\ \boldsymbol{B}(k+1) \leftarrow \boldsymbol{X}(k)$
8: **Sinon Si** $f(\boldsymbol{B}(k))f(\boldsymbol{X}(k)) < 0$ **alors**
9: $\boldsymbol{A}(k+1) \leftarrow \boldsymbol{X}(k),\ \boldsymbol{B}(k+1) \leftarrow \boldsymbol{B}(k)$
10: **Sinon**
11: $\boldsymbol{A}(k+1) \leftarrow \boldsymbol{A}(k),\ \boldsymbol{B}(k+1) \leftarrow \boldsymbol{X}(k)$
12: **Fin Si**
13: $\boldsymbol{X}(k+1) \leftarrow (\boldsymbol{A}(k+1) + \boldsymbol{B}(k+1))/2$
14: **Fin Pour**
15: $x \leftarrow \boldsymbol{X}($`kmin` $+1)$
16: **Fin Fonction**

Des codes Matlab/Octave et C correspondant de cette fonction sont données en Annexe A.4.1, respectivement en Listing A.1 et A.2. Les Listings A.3 et A.4 correspondent respectivement à un *script* Matlab et un *main* C utilisant cette fonction.

On peut noter qu'une réécriture de la méthode de dichotomie permet d'éviter d'utiliser les deux suites (a_k) et (b_k). En effet, on a

- $A = a$, $B = b$ et $x_0 = \frac{A+B}{2}$,
- $\forall k \in [\![0, k_{\min} - 1]\!]$,

$$\begin{cases} A = B = x_k & \text{si } f(x_k) = 0, \\ A = x_k,\ B \text{ inchangé} & \text{si } f(B)f(x_k) < 0, \\ B = x_k,\ A \text{ inchangé} & \text{sinon (i.e. } f(A)f(x_k) < 0.) \end{cases}$$

et

$$x_{k+1} = \frac{A+B}{2}$$

On utilise, ces formules dans l'Algorithme 3.2.

chotomie2}

Algorithme 3.2 Méthode de dichotomie : version 2

Données : $a,\ b$: deux réels $a < b$,
f : $f : [a,b] \subset \mathbb{R} \longrightarrow \mathbb{R}$ vérifiant les hypothèses de la proposition 3.1,
eps : un réel strictement positif.
Résultat : x : un réel tel que $|x - \alpha| \leqslant$ eps.

1: **Fonction** $x \leftarrow$ Dichotomie2($f, a, b,$ eps)
2: kmin $\leftarrow$ E(log((b − a)/eps)/ log(2))
3: $\boldsymbol{X} \in \mathbb{R}^{\text{kmin}+1}$ ▷ $\boldsymbol{X}(k+1)$ contiendra x_k, ...
4: A $\leftarrow a$, B $\leftarrow b$, $\boldsymbol{X}(1) \leftarrow$ (A + B)/2
5: **Pour** $k \leftarrow 1$ **à** kmin **faire**
6: **Si** $f(\boldsymbol{X}(k)) == 0$ **alors**
7: A $\leftarrow \boldsymbol{X}(k)$, B $\leftarrow \boldsymbol{X}(k)$
8: **Sinon Si** f(B)$f(\boldsymbol{X}(k)) < 0$ **alors**
9: A $\leftarrow \boldsymbol{X}(k)$ ▷ B inchangé
10: **Sinon**
11: B $\leftarrow \boldsymbol{X}(k)$ ▷ A inchangé
12: **Fin Si**
13: $\boldsymbol{X}(k+1) \leftarrow$ (A + B)/2
14: **Fin Pour**
15: $x \leftarrow \boldsymbol{X}$(kmin + 1)
16: **Fin Fonction**

Si notre objectif n'est que de calculer α_ϵ, on peut, sur le même principe, s'affranchir d'utiliser un tableau pour stocker tous les termes de la suite x_k : seul le dernier nous interesse. On propose dans l'Algorithme 3.3, une version n'utilisant pas de tableaux.

Algorithme 3.3 Méthode de dichotomie : version 3

{Algo:dichotomie

Données : $a,\ b$: deux réels $a < b$,
f : $f : [a,b] \subset \mathbb{R} \longrightarrow \mathbb{R}$ vérifiant les hypothèsesde la proposition 3.1,
eps : un réel strictement positif.
Résultat : x : un réel tel que $|x - \alpha| \leqslant$ eps.

1: **Fonction** $x \leftarrow$ Dichotomie2($f, a, b,$ eps)
2: kmin $\leftarrow$ E(log((b − a)/eps)/ log(2))
3: $A,\ B \in \mathbb{R}$
4: $A \leftarrow a,\ B \leftarrow b,\ x \leftarrow (a+b)/2$
5: **Pour** $k \leftarrow 1$ **à** kmin **faire**
6: **Si** $f(x) == 0$ **alors**
7: $A \leftarrow x,\ B \leftarrow x$
8: **Sinon Si** $f(B)f(x) < 0$ **alors**
9: $A \leftarrow x$ ▷ B inchangé
10: **Sinon**
11: $B \leftarrow x$ ▷ A inchangé
12: **Fin Si**
13: $x \leftarrow (A+B)/2$
14: **Fin Pour**
15: **Fin Fonction**

Une autre écriture est possible sans utiliser le nombre $k_{\min}$ et donc en utilisant une boucle **Tantque** (pour les anglophobes!) ou **While** (pour les anglophiles!). Celle-ci est présentée dans l'Algorithme 3.4

Algorithme 3.4 Méthode de dichotomie : version 4 {Algo:dich

Données : $a,\ b$: deux réels $a < b$,
f : $f : [a,b] \subset \mathbb{R} \longrightarrow \mathbb{R}$ ...,
eps : un réel strictement positif.
Résultat : x : un réel tel que $|x - \alpha| \leqslant$ eps.

Fonction $x \leftarrow$ Dichotomie4($f, a, b,$ eps)
  $A,\ B \in \mathbb{R}$
  $A \leftarrow a,\ B \leftarrow b,\ x \leftarrow (a+b)/2$
  Tantque $|x - A| >$ eps **faire**
    Si $f(x) == 0$ **alors**
      $A \leftarrow x,\ B \leftarrow x$
    Sinon Si $f(B)f(x) < 0$ **alors**
      $A \leftarrow x$ ▷ B inchangé
    Sinon
      $B \leftarrow x$ ▷ A inchangé
    Fin Si
    $x \leftarrow (A+B)/2$
  Fin Tantque
Fin Fonction

Que pensez-vous de l'algorithme suivant ?

Algorithme 3.5 Méthode de dichotomie : version 5 lgo:dichotomie5}

Données : $a,\ b$: deux réels $a < b$,
f : $f : [a,b] \subset \mathbb{R} \longrightarrow \mathbb{R}$
Résultat : x : un réel tel que $f(x) = 0$.

Fonction $x \leftarrow$ Dichotomie5(f, a, b)
  $A,\ B \in \mathbb{R}$
  $A \leftarrow a,\ B \leftarrow b,\ x \leftarrow (a+b)/2,$ xp $\leftarrow a$
  Tantque $x \sim=$ xp **faire**
    Si $f(B)f(x) < 0$ **alors**
      $A \leftarrow x$ ▷ B inchangé
    Sinon
      $B \leftarrow x$ ▷ A inchangé
    Fin Si
    xp $\leftarrow x$
    $x \leftarrow (A+B)/2$
  Fin Tantque
Fin Fonction

Des codes Matlab/Octave et C correspondant de cette fonction sont données en Annexe A.4.1, respectivement en Listing A.5 et A.6. Les Listings A.7 et A.8 correspondent respectivement à un *script* Matlab et un *main* C utilisant cette fonction.

3.2 Points fixes d'une fonction (dimension 1)

Soit $\Phi : [a,b] \subset \mathbb{R} \longrightarrow \mathbb{R}$ une fonction donnée. Rechercher un **point fixe** de Φ revient à

> Trouver $\alpha \in [a,b]$ tel que
> $$\alpha = \Phi(\alpha).$$

L'algorithme de la **méthode du point fixe** consiste en la construction, si elle existe, de la suite

{RSNLeq03}
$$x^{(k+1)} = \Phi(x^{(k)}),\ \forall k \in \mathbb{N} \tag{3.1}$$

avec $x^{(0)} \in [a,b]$ donné.

Definition 3.2

On dit qu'une suite $(x_k)_{k\in\mathbb{N}}$, obtenue par une méthode numérique, **converge vers α avec un ordre $p \geqslant 1$** si

$$\exists k_0 \in \mathbb{N},\ \exists C > 0 \ \text{ tels que } \ \frac{|x_{k+1}-\alpha|}{|x_k-\alpha|^p} \leqslant C,\ \forall k \geqslant k_0. \tag{3.2}$$

où $C < 1$ si $p = 1$.

On a alors le théorème suivant

Théorème 3.3: Théorème du point fixe dans $\mathbb{R}$

{RSNL:Theo:PointFix

Soient $[a,b]$ un intervalle non vide de $\mathbb{R}$ et Φ une application continue de $[a,b]$ dans lui-même. Alors, il existe **au moins** un point $\alpha \in [a,b]$ vérifiant $\Phi(\alpha) = \alpha$. Le point α est appelé **point fixe de la fonction Φ**.
De plus, si Φ est contractante (lipschitzienne de rapport $L \in [0,1[$), c'est à dire

$$\exists L < 1 \text{ t.q. } |\Phi(x)-\Phi(y)| \leqslant L|x-y|\ \forall (x,y) \in [a,b]^2, \tag{3.3}$$

{RSNL:eq:contractar

alors Φ admet un **unique** point fixe $\alpha \in [a,b]$, la suite définie en (3.1) converge vers α avec un ordre 1 pour toute donnée initiale $x^{(0)}$ dans $[a,b]$, et l'on a les deux estimations suivantes :

$$|x_k-\alpha| \leqslant L^k|x_0-\alpha|,\ \forall k \geqslant 0, \tag{3.4}$$

$$|x_k-\alpha| \leqslant \frac{L}{1-L}|x_k-x_{k-1}|,\ \forall k \geqslant 0, \tag{3.5}$$

Proof.

- On montre tout d'abord l'existence du point fixe. Pour celà, on note $f(x) = \Phi(x) - x$. f est donc une application continue de $[a,b]$ à valeurs réelles. On a $f(a) = \Phi(a) - a \geqslant 0$ et $f(b) = \Phi(b) - b \leqslant 0$ car $a \leqslant \Phi(x) \leqslant b$, pour tout $x \in [a,b]$. Si $f(a) = 0$ ou $f(b) = 0$, alors l'existence est immédiate. Sinon (i.e. $f(a) \neq 0$ et $f(b) \neq 0$), on a $f(a)f(b) < 0$ et par application directe du Théorème de Bolzano on obtient l'existence.

- On montre ensuite l'unicité sous l'hypothèse de contraction (3.3). On suppose qu'il existe α_1 et α_2 dans $[a,b]$ tels que $\Phi(\alpha_1) = \alpha_1$ et $\Phi(\alpha_2) = \alpha_2$. Dans ce cas on a

$$|\alpha_1-\alpha_2| = |\Phi(\alpha_1)-\Phi(\alpha_2)| \leqslant L|\alpha_1-\alpha_2|$$

Comme $L < 1$ on a nécessairement $\alpha_1 = \alpha_2$.

- Pour démontrer la convergence de la suite vers α, il suffit de démontrer qu'elle est de Cauchy. En effet, si elle est de Cauchy, elle converge vers une certaine limite β et Φ étant continue on a

$$\lim_{k\to+\infty} x_{k+1} = \beta = \lim_{k\to+\infty} \Phi(x_k) = \Phi(\beta).$$

Par unicité, du point fixe on obtient alors $\beta = \alpha$.
Il nous reste à montrer que la suite (x_k) est de Cauchy. Soit $k > 0$. On a

$$|x_{k+1}-x_k| = |\Phi(x_k)-\Phi(x_{k-1})| \leqslant L|x_k-x_{k-1}|,$$

et on obtient par récurrence

$$|x_{k+1}-x_k| \leqslant L^k|x_1-x_0|$$

et

$$\forall l \geqslant 0,\ |x_{k+l}-x_{k+l-1}| \leqslant L^l|x_k-x_{k-1}|.$$

Soit $p > 2$. On en déduit par application répétée de l'inégalité triangulaire que

$$|x_{k+p} - x_k| = |(x_{k+p} - x_{k+p-1}) + (x_{k+p-1} - x_{k+p-2}) + \ldots + (x_{k+1} - x_k)| = |\sum_{l=0}^{p-1} x_{k+l+1} - x_{k+l}|$$
$$\leqslant \sum_{l=0}^{p-1} |x_{k+l+1} - x_{k+l}|$$
$$\leqslant \sum_{l=0}^{p-1} L^l |x_{k+1} - x_k| = \frac{1-L^p}{1-L}|x_{k+1} - x_k| \quad \textit{(voir somme partielle d'une série géométrique)}$$
$$\leqslant \frac{1-L^p}{1-L} L^k |x_1 - x_0|.$$

Comme $L^k \to 0$ quand $k \to +\infty$, on conclu que (x_k) est une suite de Cauchy.

On a $\forall k \geqslant 1$,

$$|x_{k+1} - \alpha| = |\Phi(x_k) - \Phi(\alpha)| \leqslant L|x_k - \alpha|$$

Comme $L < 1$, on a immediatement l'ordre 1 (au moins).

- Pour démontrer la première estimation, on note que, $\forall k \geqslant 1$,

$$|x_k - \alpha| = |\Phi(x_{k-1}) - \Phi(\alpha)| \leqslant L|x_{k-1} - \alpha|$$

et donc par récurrence

$$|x_k - \alpha| \leqslant L^k |x_0 - \alpha|.$$

Pour la seconde, on note que, $\forall k \geqslant 1$, $\forall p \geqslant 1$,

$$|x_{k+p} - x_k| \leqslant \frac{1-L^p}{1-L}|x_{k+1} - x_k| \leqslant \frac{1-L^p}{1-L} L|x_k - x_{k-1}|$$

et en faisant tendre p vers l'infini on obtient le résultat souhaité.

□

{PointFixeGlobale}

Théorème 3.4: Convergence globale de la méthode du point fixe

Soit $\Phi \in \mathcal{C}^1([a,b])$ vérifiant $\Phi([a,b]) \subset [a,b]$ et

$$\exists L < 1 \text{ tel que } \forall x \in [a,b],\ |\Phi'(x)| \leqslant L, \tag{3.6}$$

Soit $x_0 \in [a,b]$ et $(x_k)_{k\in\mathbb{N}}$ la suite définie par $x_{k+1} = \Phi(x_k)$. On a alors

1. la fonction Φ admet un unique point fixe $\alpha \in [a,b]$,
2. $\forall k \in \mathbb{N},\ x_k \in [a,b]$,
3. la suite (x_k) converge vers α avec un ordre 1 au moins.
4. $$\lim_{k\to+\infty} \frac{x_{k+1} - \alpha}{x_k - \alpha} = \Phi'(\alpha). \tag{3.7}$$

Proof. Comme Φ est continue sur $[a,b]$ et vérifie $\Phi([a,b]) \subset [a,b]$, le théoréme 3.3 (du point fixe) assure l'existence d'un point fixe. Pour l'unicité, on note $\alpha_1 \in [a,b]$ et $\alpha_2 \in [a,b]$, deux points fixes de Φ. On a donc $\Phi(\alpha_1) = \alpha_1$ et $\Phi(\alpha_2) = \alpha_2$. D'après le théorème des accroissements finis il existe $\xi \in]\min(\alpha_1,\alpha_2), \max(\alpha_1,\alpha_2)[\subset]a,b[$ tel que

$$\Phi(\alpha_1) - \Phi(\alpha_2) = (\alpha_1 - \alpha_2)\Phi'(\xi).$$

Comme $|\Phi'(\xi)| < 1$, on obtient $\alpha_1 - \alpha_2 = 0$, i.e. $\alpha_1 = \alpha_2$.

Pour le second point, on a immediatement $\forall k \in \mathbb{N}$, $x_k \in [a,b]$ car $x_0 \in [a,b]$ et $\Phi([a,b]) \subset [a,b]$. Le troisième point découle du théorème des accroissements. En effet, pour tout $k > 0$, il existe $\xi_k \in]\min(\alpha, x_{k-1}), \max(\alpha, x_{k-1}) \subset]a,b[$ tel que

$$\Phi(x_{k-1}) - \Phi(\alpha) = (x_{k-1} - \alpha)\Phi'(\xi_k).$$

On obtient alors

$$|x_k - \alpha| = |\Phi(x_{k-1}) - \Phi(\alpha)| \leqslant L|x_{k-1} - \alpha|$$

et donc par récurrence

$$|x_k - \alpha| \leqslant L^k |x_0 - \alpha|.$$

Comme $L < 1$, on obtient

$$\lim_{k \to +\infty} |x_k - \alpha| = 0.$$

□ 1

Théorème 3.5: Convergence locale de la méthode du point fixe

{RSNL:Theo:Ostrowsk

Soit α un point fixe d'une fonction Φ de classe $\mathcal{C}^1$ au voisinage de α.
Si $|\Phi'(\alpha)| < 1$, alors il existe $\delta > 0$ pour lequel x_k converge vers α pour tout x_0 tel que $|x_0 - \alpha| < \delta$.
De plus, on a

$$\lim_{k \to +\infty} \frac{x_{k+1} - \alpha}{x_k - \alpha} = \Phi'(\alpha). \tag{3.8}$$

{RSNL:eq:Ostrowski}
2

Proof. Comme Φ est continûment différentiable dans un voisinage de α avec $|\Phi'(\alpha)| < 1$, alors il existe $\delta > 0$ tel que $\forall x \in \mathcal{V} = [\alpha - \delta, \alpha + \delta]$, $|\Phi'(x)| < 1$.
Soient x et y appartenant à $\mathcal{V}$, d'après le théorème des accroissements finis il existe $\xi \in]\alpha - \delta, \alpha + \delta[$ tel que

$$f(x) - f(y) = (x - y)\Phi'(\xi).$$

Ceci entraine que Φ est contractante sur $\mathcal{V}$. De plus $\Phi(\mathcal{V}) \subset \mathcal{V}$ car $\forall x \in \mathcal{V}$

$$|\Phi(x) - \Phi(\alpha)| = |x - \alpha||\Phi'(\xi)| \leqslant \delta |\Phi'(\xi)| \leqslant \delta$$

et donc $|\Phi(x) - \alpha| \leqslant \delta$ i.e. $\Phi(x) \in \mathcal{V}$. On peut donc appliquer le théorème du point fixe (Théorème 3.3) (avec $[a,b] = [\alpha - \delta, \alpha + \delta]$)) ce qui donne la convergence de (x_k) vers α pour tout $x_0 \in \mathcal{V}$.
Pour demontrer (3.8), on utilise une nouvelle fois le théorème des accroissements finis : il existe $\xi_k \in]\min(x_k, \alpha), \max(x_k, \alpha)[$ tel que

$$\Phi(x_k) - \Phi(\alpha) = \Phi'(\xi_k)(x_k - \alpha).$$

Comme $\Phi(\alpha) = \alpha$ et $\Phi(x_k) = x_{k+1}$, on obtient

$$\frac{x_{k+1} - \alpha}{x_k - \alpha} = \Phi'(\xi_k).$$

Quand $k \to +\infty$, on a $x_k \to \alpha$ et donc $\xi_k \to \alpha$. Par continuité de la fonction Φ' on obtient (3.8). □ 3

Proposition 3.6

{RSNL:Proposition:c

Soit $p \in \mathbb{N}^*$, et $\Phi \in \mathcal{C}^{p+1}(\mathcal{V})$ pour un certain voisinage $\mathcal{V}$ de α point fixe de Φ. Si $\Phi^{(i)}(\alpha) = 0$, pour $1 \leqslant i \leqslant p$ et si $\Phi^{(p+1)}(\alpha) \neq 0$, alors la méthode de point fixe associée à la fonction Φ est d'ordre $p+1$ et

$$\lim_{k \to +\infty} \frac{x_{k+1} - \alpha}{(x_k - \alpha)^{p+1}} = \frac{\Phi^{(p+1)}(\alpha)}{(p+1)!}. \tag{3.9}$$

{RSNL:eq:ordrep}
4

Proof. Les hypothèses du théorème 3.5 étant vérifiées ($\Phi'(\alpha)=0$), la suite (x_k) converge vers α. D'après un développement de Taylor de Φ en α, il existe η_k entre x_k et α vérifiant

$$\Phi(x_k)=\sum_{i=0}^{p}\frac{(x_k-\alpha)^i}{i!}\Phi^{(i)}(\alpha)+\frac{(x_k-\alpha)^{p+1}}{(p+1)!}\Phi^{(p+1)}(\eta_k).$$

Comme $\alpha=\Phi(\alpha)$ et $\Phi^{(i)}(\alpha)=0$, pour $1\leqslant i\leqslant p$ on obtient

$$\begin{aligned}x_{k+1}&=\Phi(\alpha)+\sum_{i=1}^{p}\frac{(x_k-\alpha)^i}{i!}\Phi^{(i)}(\alpha)+\frac{(x_k-\alpha)^{p+1}}{(p+1)!}\Phi^{(p+1)}(\eta_k)\\&=\alpha+\frac{(x_k-\alpha)^{p+1}}{(p+1)!}\Phi^{(p+1)}(\eta_k)\end{aligned}$$

De plus (η_k) converge vers α car η_k est entre x_k et α et donc comme $\Phi^{(p+1)}$ est continue au voisinage de α on obtient

$$\lim_{k\to+\infty}\frac{x_{k+1}-\alpha}{(x_k-\alpha)^{p+1}}=\lim_{k\to+\infty}\frac{\Phi^{(p+1)}(\eta_k)}{(p+1)!}=\frac{\Phi^{(p+1)}(\alpha)}{(p+1)!}.$$

□

3.2.1 Points fixes attractifs et répulsifs

Soit $\Phi:[a,b]\longrightarrow[a,b]$ une application de classe $\mathcal{C}^1$ admettant un point fixe $\alpha\in[a,b]$.

Points fixes attractifs

On suppose $|\Phi'(\alpha)|<1$. D'après le théorème 3.5, il existe $\delta>0$ tel que

$$\forall x_0\in[\alpha-h,\alpha+\delta],\quad\lim_{k\to+\infty}x_k=\alpha.$$

Dans ce cas on dit que α est un **point fixe attractif** pour Φ.

Exercice 3.2.1

{RSNL:exo101}

On suppose de plus que $\Phi'(\alpha)=0$ et $\exists M\in\mathbb{R}^+$ tel que $|\Phi''(x)|\leqslant M$ pour tout $x\in[\alpha-h,\alpha+\delta]$.

Q. 1 *1. Montrer que*

$$\forall x_0\in[\alpha-h,\alpha+\delta],\ |x_k-\alpha|\leqslant\frac{2}{M}\left(\frac{1}{2}M|x_0-\alpha|\right)^{2^k}$$

2. Quel est l'ordre de convergence dans ce cas.

Q. 2 *A quelle condition a-t'on*

$$|x_k-\alpha|\leqslant\frac{2}{M}10^{2^k}.$$

Correction Exercice 3.2.1

Q. 1 1. D'après la formule de Taylor, on a $\exists\eta\in]\min(\alpha,x),\max(\alpha,x)[$ tel que

$$\begin{aligned}\Phi(x)&=\Phi(\alpha)+(x-a)\Phi'(\alpha)+\frac{(x-\alpha)^2}{2!}\Phi''(\eta)\\&=\alpha+\frac{1}{2}\Phi''(\eta)(x-\alpha)^2.\end{aligned}$$

On en déduit que $|\Phi(x)-\alpha|\leqslant\frac{M}{2}|x-\alpha|^2$ ce qui s'écrit encore $\frac{M}{2}|\Phi(x)-\alpha|\leqslant(\frac{M}{2}|x-\alpha|)^2$. Or si $x_0\in[\alpha-h,\alpha+\delta]$, alors $x_{k-1}\in[\alpha-h,\alpha+\delta]$, $\forall k\in\mathbb{N}^*$. On a alors

$$\frac{M}{2}|\Phi(x_{k-1})-\alpha|=\frac{M}{2}|x_k-\alpha|\leqslant(\frac{M}{2}|x_{k-1}-\alpha|)^2$$

et donc par récurrence

$$\frac{M}{2}|x_k-\alpha|\leqslant(\frac{M}{2}|x_0-\alpha|)^{2^k}.$$

2. On a

$$|x_k - \alpha| \leqslant \frac{2}{M}(\frac{M}{2}|x_{k-1} - \alpha|)^2$$

c'est à dire

$$\frac{|x_k - \alpha|}{|x_{k-1} - \alpha|^2} \leqslant \frac{M}{2}$$

et donc la méthode est d'ordre 2.

Q. 2 En supposant de plus que $|x_0 - \alpha| \leqslant \frac{1}{5M}$, on obtient immédiatement le résultat.

◇

Points fixes répulsifs

Cas $|\Phi'(\alpha)| > 1$. Par définition de la dérivée en α on a

$$\lim_{\substack{x \to \alpha \\ x \neq \alpha}} \left| \frac{\Phi(x) - \Phi(\alpha)}{x - \alpha} \right| = |\Phi'(\alpha)| > 1.$$

Il existe alors $\delta > 0$ tel que

$$\forall x \in [\alpha - h, \alpha + \delta] \backslash \{\alpha\}, \quad |\Phi(x) - \alpha| > |x - \alpha|$$

et donc la suite (x_k) ne peut converger vers α et ceci même si x_0 est très proche de α. Dans ce cas on dit que α est un **point fixe répulsif** pour Φ.
Toutefois, on *peut rattrapper le coup*. En effet, comme Φ' est non nulle et de signe constant au voisinage de α : il existe $h > 0$ tel que la fonction Φ soit strictement monotone sur $I = [\alpha - h, \alpha + h]$.. D'après le corollaire A.4 Φ est une bijection de I dans l'intervalle fermé $J = \Phi^{-1}(I)$. Comme $\alpha = \Phi(\alpha)$, on a $\alpha \in J$ et $\Phi^{-1}(\alpha) = \alpha$. Le problème de point fixe trouver $x \in I$ tel que $\Phi(x) = x$ revient alors à trouver $x \in J$ tel que $\Phi^{-1}(x) = x$. Or $\Phi'(\alpha) \neq 0$ et $\Phi(\alpha) = \alpha$, la proposition A.5 donne alors $(\Phi^{-1})'(\alpha) = 1/\Phi'(\alpha)$ et donc $|(\Phi^{-1})'(\alpha)| < 1$. Le point α est un **point fixe attractif** pour Φ^{-1}.

Points fixes indéterminés

Dans le cas $|\Phi'(\alpha)| = 1$, on ne peut conclure dans le cas général.

3.2.2 Interprétations graphiques de la méthode du point fixe

Exemple 1 : point fixe de la fonction $x \mapsto x^2$.

On s'interesse ici au point fixe $\alpha = 1$ de la fonction $\Phi : x \mapsto x^2$.

Comme $\Phi'(\alpha) = 2$ le point fixe α est répulsif. On donne dans le tableau suivant les premiers termes de deux suites : l'une avec $x_0 = 1.05$ et l'autre avec $x_0 = 0.95$. Dans ce dernier cas, la suite va converger ... vers un autre point fixe.

k	x_k	x_k
0	1.05000	0.950000
1	1.10250	0.902500
2	1.21551	0.814506
3	1.47746	0.663420
⋮	⋮	⋮
8	265742.	1.98264×10^{-6}

Les premières itérations de ces deux suites sont représentées en Figure 3.4.

Sur l'intervalle $[0, +\infty[$, la fonction Φ admet comme fonction inverse $\Phi^{-1}(x) = \sqrt{x}$ et on a aussi $\Phi^{-1}(1) = 1$ (i.e. $\alpha = 1$ est un point fixe de Φ^{-1}). De plus $(\Phi^{-1})'(1) = 1/2 < 1$, ce qui permet d'affirmer que $\alpha = 1$ est un point fixe **attractif** de Φ^{-1}). On donne dans le tableau suivant les premiers termes de

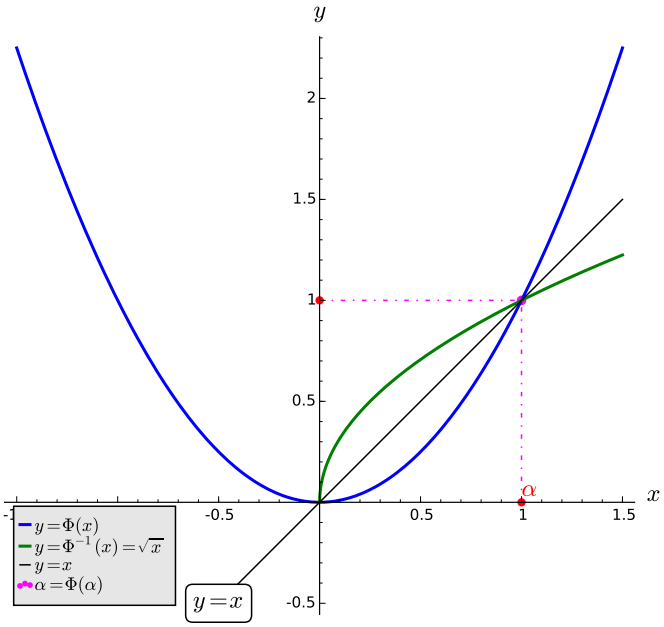
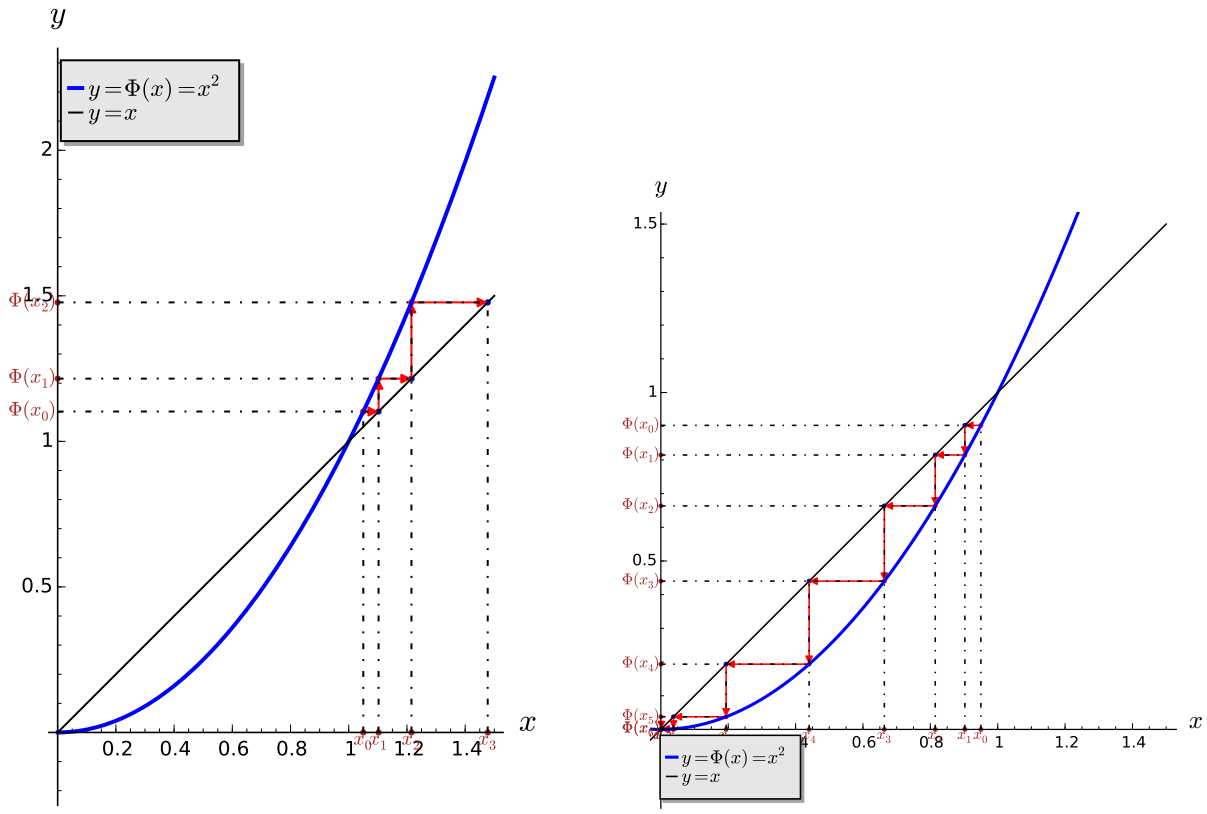


Figure 3.3: fonction x^2 et son fonction inverse $\sqrt{x}$ sur $[0,+\infty[$ RSNL:fig:pointfixe:01



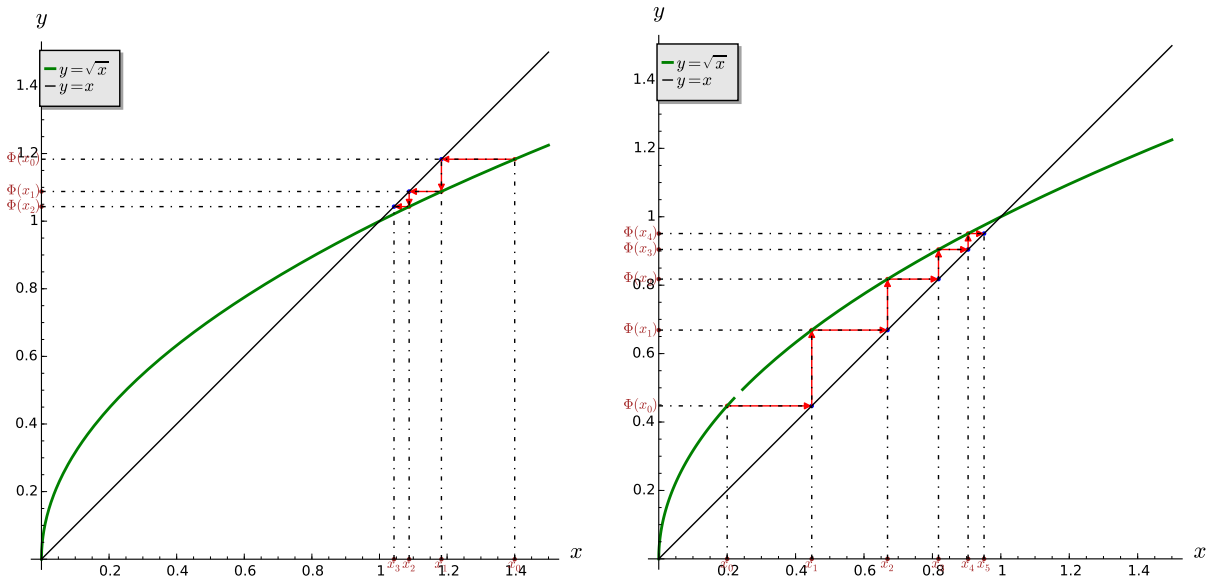
(a) $x_0 = 1.05$ (b) $x_0 = 0.950$

Figure 3.4: $\alpha = 1$, point fixe répulsif de $x \mapsto x^2$ RSNL:pointfixe:repulsif01

deux suites : l'une avec $x_0 = 1.40$ et l'autre avec $x_0 = 0.200$.

k	x_k	x_k
0	1.40000	0.200000
1	1.18322	0.447214
2	1.08776	0.668740
3	1.04296	0.817765
⋮	⋮	⋮
8	1.00132	0.993733

Les premières itérations de ces deux suites sont représentées en Figure 3.5.



(a) $x_0 = 1.40$ (b) $x_0 = 0.200$

Figure 3.5: $\alpha = 1$, point fixe attractif de $x \mapsto \sqrt{x}$ RSNL:pointfixe:attractif01

Exemple 2 : point fixe de la fonction $x \mapsto x^2 - x + 1$.

On s'interesse ici au point fixe $\alpha = 1$ de la fonction $f : x \mapsto x^2 - x + 1$ représenté en Figure 3.6. On note que $f'(\alpha) = 1$.

On donne dans le tableau suivant les premiers termes de deux suites : l'une avec $x_0 = 1.10$ diverge et l'autre avec $x_0 = 0.500$ converge vers α.

k	x_k	x_k
0	1.10000	0.500000
1	1.11000	0.750000
2	1.12210	0.812500
3	1.13701	0.847656
⋮	⋮	⋮
8	1.32397	0.918223

Les premières itérations de ces deux suites sont représentées en Figure 3.7.

Exemple 3 : point fixe de fonctions affines particulières

On va étudier les points fixes des trois fonctions affines vérifiant $f_1(1) = f_2(1) = f_3(1) = 1$ et $f_1'(1) = -1$, $f_2'(1) = -3/4$ et $f_3'(1) = -4/3$. Ces trois fonctions sont données par $f_1(x) = -x + 2$, $f_2(x) = -\frac{3}{4}x + \frac{7}{4}$ et $f_3(x) = -\frac{4}{3}x + \frac{7}{3}$.

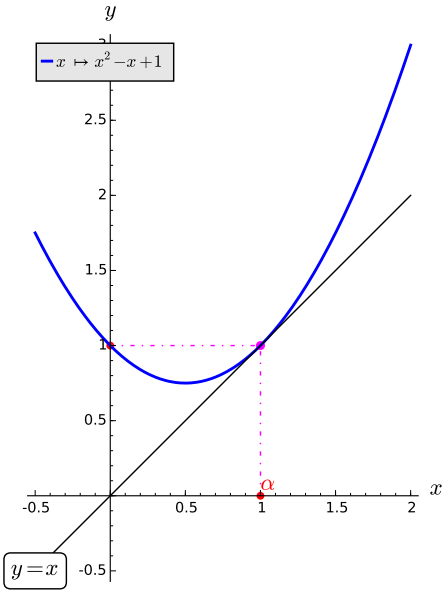
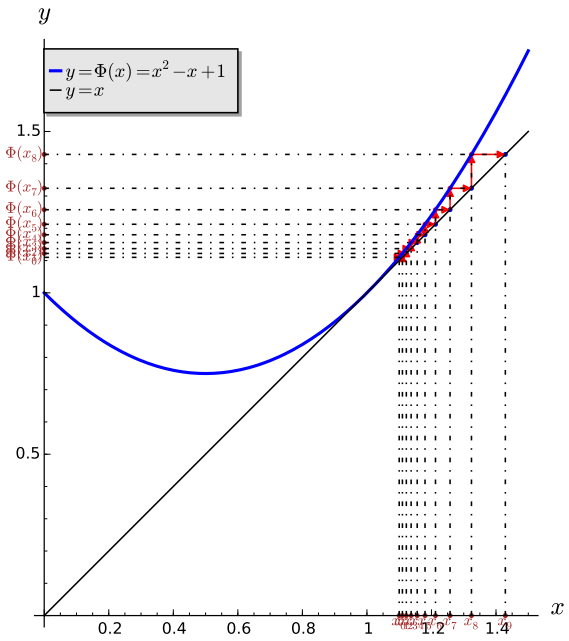
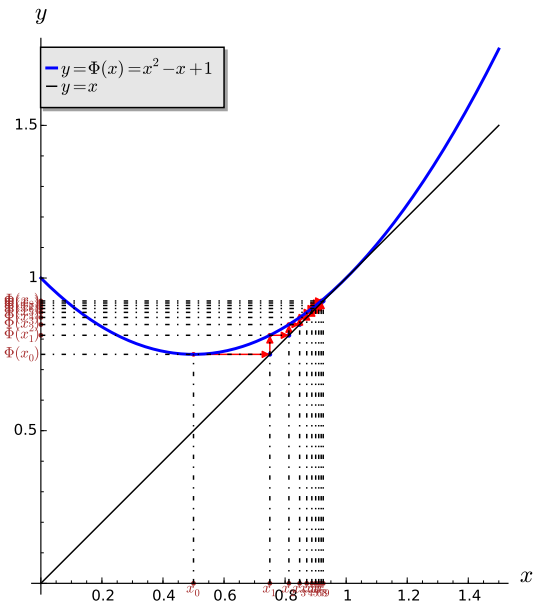


Figure 3.6: fonction $x^2 - x + 1$ et son point fixe $\alpha = 1$. RSNL:fig:pointfixe:02



(a) $x_0 = 1.1$, α point répulsif



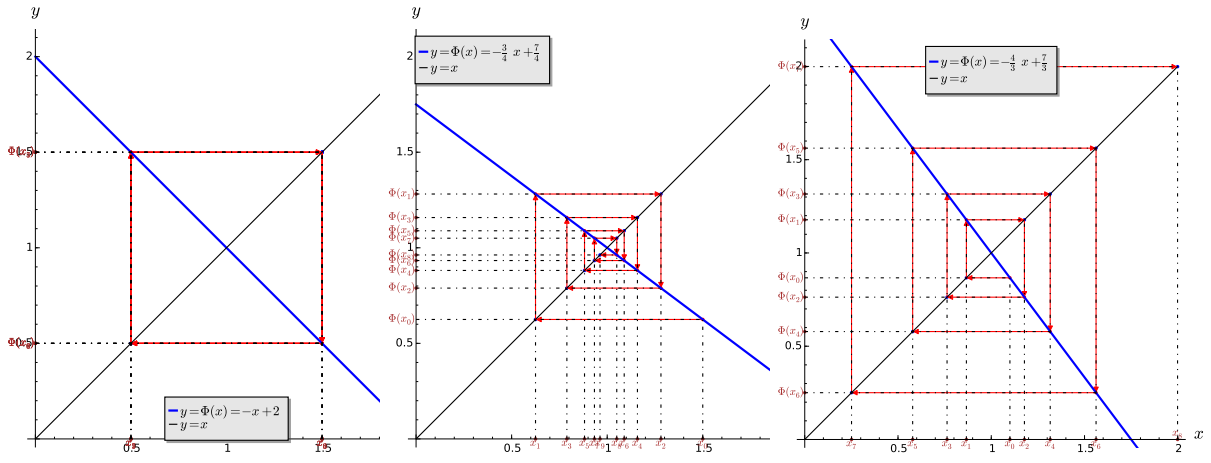
(b) $x_0 = 0.50$, α point attractif

Figure 3.7: $\alpha = 1$, point fixe attractif ou répulsif de $x \mapsto x^2 - x + 1$ RSNL:pointfixe:repulsifattrac

Comme $\Phi'(\alpha) = 2$ le point fixe α est répulsif. On donne dans le tableau suivant les premiers termes de trois suites obtenues avec $x_0 = 1.5$ pour les suites associées à f_1 et f_2, et avec $x_0 = 1.1$ pour celle associée à f_2.

k	x_k (f_1)	x_k (f_2)	x_k (f_3)
0	1.50000	1.50000	1.10000
1	0.500000	0.625000	0.866667
2	1.50000	1.28125	1.17778
3	0.500000	0.789062	0.762963
⋮	⋮	⋮	
19	0.500000	0.997886	−22.6503
20	1.50000	1.00159	32.5337

Les premières itérations des trois suites obtenues sont représentées en Figure 3.8.



(a) $x_0 = 1.50$ (b) $x_0 = 1.50$ (c) $x_0 = 1.10$

Figure 3.8: $\alpha = 1$, point fixe de fonctions affines particulières RSNL:pointfixe:Affine01

3.2.3 Algorithme générique du point fixe

Le principe de base est très simple et donné par les Algorithmes 3.6 et 3.7, écrit respectivement avec une boucle **Tantque** et une boucle **Répéter**.

Algorithme 3.6 Méthode de point fixe : version **Tantque** *formel*

1: $k \leftarrow 0$
2: **Tantque** non convergence **faire**
3: $x_{k+1} \leftarrow \Phi(x_k)$
4: $k \leftarrow k + 1$
5: **Fin Tantque**
6: $\alpha_\epsilon \leftarrow x_k$ ▷ le dernier calculé.

Algorithme 3.7 Méthode de point fixe : version **Répéter** *formel*

{Algo:pointfixe1}

1: $k \leftarrow 0$
2: **Répéter**
3: $k \leftarrow k + 1$
4: $x_k \leftarrow \Phi(x_{k-1})$
5: **jusqu'à** convergence
6: $\alpha_\epsilon \leftarrow x_k$ ▷ le dernier calculé.

Nous allons plus particulièrement étudier les critères d'arrêt des boucles **Tantque** et **Répéter** (négation l'un de l'autre) qui sont bien trop flous : on s'arrête quand on converge!

Tout d'abord, on n'est pas sur de converger : il faudra donc n'autoriser qu'un nombre maximum d'itérations `kmax`. De plus, si l'on converge vers une certaine valeur α celle-ci n'étant pas connue à l'avance (sinon l'algo n'a aucun intérêt), on ne peut utiliser, comme condition du **Tantque**, $|x_k - \alpha| > \epsilon$ ($|x_k - \alpha| \leqslant \epsilon$ pour la condition du **Répéter**) où ϵ serait la précision souhaitée. L'idée serait de comparer x_{k+1} et x_k et donc de tester s'ils sont suffisament proches : la condition serait alors $|\Phi(x_k) - x_k| = |x_{k+1} - x_k| >$ `tol`, (boucle **Tantque**) ou $|\Phi(x_k) - x_k| \leqslant$ `tol`, (boucle **Répéter**) avec `tol` la tolérance souhaitée. Il faut noter que dans ce cas la valeur `tol` doit être choisie correctement et *dépend* de l'ordre

de grandeur de α. Si α est de l'ordre de 10^8, `tol` $= 1$ comme valeur est raisonnable. Toutefois on peut lui préférer une condition *relative* $\frac{|\Phi(x_k)-x_k|}{|x_k|+1} >$ `tol` (boucle **Tantque**) ou $\frac{|\Phi(x_k)-x_k|}{|x_k|+1} \leqslant$ `tol` (boucle **Répéter**) qui permet à `tol` de correspondre à une tolérance *relative* souhaitée que ce soit pour de grandes valeurs de α ou pour des petites. Les algorithmes du point fixe (recherche de α solution de $\Phi(x) = x$) avec notations mathématiques sont donnés par Algorithme 3.8 et 3.9, respectivement avec des boucles **Tantque** et **Répéter**.

{Algo:pointfixe0a}

Algorithme 3.8 Méthode de point fixe : version **Tantque** *formel* avec critères d'arrêt

{Algo:pointfixe1a}

$k \leftarrow 0$
$\texttt{err} \leftarrow |\Phi(x_0) - x_0|$ ▷ ou $\frac{|\Phi(x_0)-x_0|}{|x_0|+1}$
Tantque $\texttt{err} > \epsilon$ et $k \leqslant \texttt{kmax}$ **faire**
  $k \leftarrow k + 1$
  $x_k \leftarrow \Phi(x_{k-1})$
  $\texttt{err} \leftarrow |\Phi(x_k) - x_k|$ ▷ ou $\frac{|\Phi(x_k)-x_k|}{|x_k|+1}$
Fin Tantque
Si $\texttt{err} \leqslant \texttt{tol}$ **alors** ▷ Convergence
  $\alpha_{\texttt{tol}} \leftarrow x_k$ ▷ $|\Phi(\alpha_{\texttt{tol}}) - \alpha_{\texttt{tol}}| \leqslant \texttt{tol}$
Fin Si

Algorithme 3.9 Méthode de point fixe : version **Répéter** *formel* avec critères d'arrêt

$k \leftarrow 0$
Répéter
  $\texttt{err} \leftarrow |\Phi(x_k) - x_k|$ ▷ ou $\frac{|\Phi(x_k)-x_k|}{|x_k|+1}$
  $x_{k+1} \leftarrow \Phi(x_k)$
  $k \leftarrow k + 1$
jusqu'à $\texttt{err} \leqslant \texttt{tol}$ ou $k > \texttt{kmax}$
Si $\texttt{err} \leqslant \texttt{tol}$ **alors** ▷ Convergence
  $\alpha_{\texttt{tol}} \leftarrow x_k$ ▷ $|\Phi(\alpha_{\texttt{tol}}) - \alpha_{\texttt{tol}}| \leqslant \texttt{tol}$
Fin Si

Des versions, sous forme de fonction, plus proches de la programmation sont proposées en Algorithme 3.10 et 3.11. Bien évidemment, suivant l'usage que l'on souhaite de ces fonctions, il est facile de les adapter pour retourner plus d'informations (nombre d'itération effectives, statut de convergence, ...)

{Algo:pointfixe0b}

Algorithme 3.10 Méthode de point fixe : version **Tantque** avec critères d'arrêt

{Algo:pointfixe1b}

Données :
Φ : $\Phi : \mathbb{R} \longrightarrow \mathbb{R}$,
x_0 : donnée initiale, $x_0 \in \mathbb{R}$,
`tol` : la tolérence, $\texttt{tol} \in \mathbb{R}^+$,
`kmax` : nombre maximum d'itérations, $\texttt{kmax} \in \mathbb{N}^*$
Résultat :
$\alpha_{\texttt{tol}}$: un réel tel que $|\Phi(\alpha_{\texttt{tol}}) - \alpha_{\texttt{tol}}| \leqslant \texttt{tol}$ (ou $\frac{|\Phi(\alpha_{\texttt{tol}})-\alpha_{\texttt{tol}}|}{|\alpha_{\texttt{tol}}|+1} \leqslant \texttt{tol}$)

Fonction $\alpha_{\texttt{tol}} \leftarrow$ PtFixe($\Phi, x_0, \texttt{tol}, \texttt{kmax}$)
  $k \leftarrow 0$, $\alpha_{\texttt{tol}} \leftarrow \varnothing$
  $\texttt{x} \leftarrow x_0$, $\texttt{fx} \leftarrow \Phi(x_0)$,
  $\texttt{err} \leftarrow |\texttt{fx} - \texttt{x}|$ ▷ ou $\frac{|\texttt{fx}-\texttt{x}|}{|\texttt{x}|+1}$
  Tantque $\texttt{err} > \texttt{tol}$ et $k \leqslant \texttt{kmax}$ **faire**
    $k \leftarrow k + 1$
    $\texttt{x} \leftarrow \texttt{fx}$
    $\texttt{fx} \leftarrow \Phi(\texttt{x})$
    $\texttt{err} \leftarrow |\texttt{fx} - \texttt{x}|$ ▷ ou $\frac{|\texttt{fx}-\texttt{x}|}{|\texttt{x}|+1}$
  Fin Tantque
  Si $\texttt{err} \leqslant \texttt{tol}$ **alors** ▷ Convergence
    $\alpha_{\texttt{tol}} \leftarrow \texttt{x}$
  Fin Si
Fin Fonction

Algorithme 3.11 Méthode de point fixe : version **Répéter** avec critères d'arrêt

Données :
Φ : $\Phi : \mathbb{R} \longrightarrow \mathbb{R}$,
x_0 : donnée initiale, $x_0 \in \mathbb{R}$,
`tol` : la tolérence, $\texttt{tol} \in \mathbb{R}^+$,
`kmax` : nombre maximum d'itérations, $\texttt{kmax} \in \mathbb{N}^*$
Résultat :
$\alpha_{\texttt{tol}}$: un réel tel que $|\Phi(\alpha_{\texttt{tol}}) - \alpha_{\texttt{tol}}| \leqslant \texttt{tol}$ (ou $\frac{|\Phi(\alpha_{\texttt{tol}})-\alpha_{\texttt{tol}}|}{|\alpha_{\texttt{tol}}|+1} \leqslant \texttt{tol}$)

Fonction $\alpha_{\texttt{tol}} \leftarrow$ PtFixe($\Phi, x_0, \texttt{tol}, \texttt{kmax}$)
  $k \leftarrow 0$, $\alpha_{\texttt{tol}} \leftarrow \varnothing$
  $\texttt{x} \leftarrow x_0$
  Répéter
    $\texttt{xp} \leftarrow \texttt{x}$
    $\texttt{x} \leftarrow \Phi(\texttt{xp})$
    $\texttt{err} \leftarrow |\texttt{x} - \texttt{xp}|$ ▷ ou $\frac{|\texttt{x}-\texttt{xp}|}{|\texttt{xp}|+1}$
    $k \leftarrow k + 1$
  jusqu'à $\texttt{err} \leqslant \texttt{tol}$ ou $k > \texttt{kmax}$
  Si $\texttt{err} \leqslant \texttt{tol}$ **alors** ▷ Convergence
    $\alpha_{\texttt{tol}} \leftarrow \texttt{x}$
  Fin Si
Fin Fonction

3.2.4 Méthodes de points fixes pour la recherche de racines

On peut noter que résoudre $f(x) = 0$ revient, par exemple, à résoudre $\Phi(x) := x + f(x) = x$. De manière plus générale, si $\mathcal{F}$ est une fonction continue vérifiant $\mathcal{F}(0) = 0$ alors $\Phi(x) = x + \mathcal{F}(f(x))$ est un autre choix possible.

Soit f une fonction de classe $\mathcal{C}^1$ dans un voisinage d'une de ses racines simple α.

Selectionner une version :

Version 1 En appliquant la formule de Taylor pour tout x dans ce voisinage, il existe $\xi \in]\min(x,\alpha), \max(x,\alpha)[$ tel que

$$f(\alpha) = 0 = f(x) + (\alpha - x)f'(\xi).$$

Ceci conduit à la méthode itérative suivante : x_0 étant donné dans le voisinage de α, on doit, $\forall k \in N$, déterminer x_{k+1} vérifiant $f(x_k) + (x_{k+1} - x_k)q_k = 0$ sachant que q_k est une approximation de $f'(\xi)$.

Version 2 On cherche à déterminer une méthode itérative permettant de calculer x_{k+1} en fonction des valeurs précédentes en espérant que $|x_{k+1} - \alpha| \leqslant |x_k - \alpha|$. Pour celà, on écrit une formule de taylor en x_k supposé proche de α et on note $h = \alpha - x_k$

$$f(\alpha) = 0 = f(x_k) + hf'(x_k) + o(h)$$

L'idéal serait de trouver la valeur exacte de h et de prendre $x_{k+1} = x_k + h$ mais ceci n'est pas possible dans un cadre général. C'est pourquoi on cherche une approximation $\tilde{h}$ de h. De plus, on ne connait pas forcément explicitement la dérivée de f. On note donc q_k une approximation de $f'(x_k)$. On choisi alors $\tilde{h}$ comme solution de

$$f(x_k) + \tilde{h}q_k = 0$$

Si $q_k \neq 0$, on obtient la suite itérative

$$x_{k+1} = x_k - \frac{f(x_k)}{q_k}, \ \forall k \in \mathbb{N} \tag{3.10}$$

La valeur x_{k+1} est de fait l'intersection de la droite passant par le point $((x_k), f(x_k))$ et de pente q_k avec l'axe des x.

Par la suite, on va écrire un algorithme du point fixe et étudier différentes méthodes :

- la **méthode de la corde** :
$$q_k = q = \frac{f(b) - f(a)}{b - a}$$
- la **méthode de la sécante** :
$$q_k = \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$$
où x_{-1} et x_0 sont données,
- la **méthode de Newton** : en supposant f' connu, on prend
$$q_k = f'(x_k).$$

La méthode de la corde

Soit f une fonction de classe $\mathcal{C}^1$ sur $[a,b]$ tel que $f(a) \neq f(b)$. Soit $x_0 \in [a,b]$ donné. La suite obtenue par la méthode de la corde est donnée par

$$x_{k+1} = x_k - \frac{b-a}{f(b) - f(a)} f(x_k), \ \forall k \in \mathbb{N}. \tag{3.11}$$ {RSNL:corde:sui

On peut voir cette relation comme un cas particulier de la méthode du point fixe. En effet, en prenant $\Phi(x) = x - \frac{b-a}{f(b)-f(a)} f(x)$, la suite définie en (3.11) s'écrit $x_{k+1} = \Phi(x_k)$.

On a alors le résultat suivant

Proposition 3.7: convergence de la méthode de la corde

{RSNL:Corde01

Soit $f \in \mathcal{C}^1([a,b])$ tel que $f(b) \neq f(a)$ et $\lambda = \frac{b-a}{f(b)-f(a)}$. On note $(x_k)_{k\in\mathbb{N}}$ la suite définie par $x_0 \in [a,b]$ et pour tout $k \geqslant 0$

$$x_{k+1} = x_k - \lambda f(x_k). \tag{3.12}$$ {RSNL:Corde01

On suppose de plus que $\forall x \in [a,b]$

$$\min(\lambda(x-a), \lambda(x-b)) \leqslant f(x) \leqslant \max(\lambda(x-a), \lambda(x-b)) \tag{3.13}$$ {RSNL:Corde01

$$\min(0, 2\lambda) < f'(x) < \max(0, 2\lambda) \tag{3.14}$$ {RSNL:Corde01

alors la suite (x_k) converge vers l'unique racine $\alpha \in [a,b]$ de f.

Proof. voir correction Exercice 3.2.2 □

Exercice 3.2.2

{RSNL:exo103}

Soit f une fonction de classe $\mathcal{C}^1$ sur $[a,b]$ vérifiant $f(a)f(b) < 0$. et $\lambda = \frac{b-a}{f(b)-f(a)}$. Soit $x_0 \in [a,b]$ donné. La suite obtenue par la méthode de la corde est donnée par

$$x_{k+1} = x_k - \frac{b-a}{f(b)-f(a)} f(x_k), \ \forall k \in \mathbb{N}.$$

On note $\Phi(x) = x - \frac{b-a}{f(b)-f(a)} f(x)$.

{exo:corde:Q1}

Q. 1 *Montrer que si pour tout $x \in [a,b]$ on a*

{RSNL:exo:corde01}

$$\min(\lambda(x-a), \lambda(x-b)) \leqslant f(x) \leqslant \max(\lambda(x-a), \lambda(x-b)) \tag{3.15}$$

alors $\Phi([a,b]) \subset [a,b]$.

{exo:corde:Q1}

Q. 2 *Montrer que si pour tout $x \in [a,b]$ on a*

{RSNL:exo:corde02}

$$\min(0, 2\lambda) < f'(x) < \max(0, 2\lambda) \tag{3.16}$$

alors $|\Phi'(x)| < 1$.

Q. 3 *En déduire que sous les deux conditions précédentes la méthode de la corde converge vers l'unique solution $\alpha \in [a,b]$ de $f(x) = 0$.*

Correction Exercice 3.2.2

Q. 1 Si $\lambda > 0$, l'inéquation (3.15) devient

$$\begin{aligned} \lambda(x-b) \leqslant f(x) \leqslant \lambda(x-a) &\Leftrightarrow a \leqslant x - \frac{f(x)}{\lambda} \leqslant b \\ &\Leftrightarrow a \leqslant \Phi(x) \leqslant b. \end{aligned}$$

Si $\lambda < 0$, l'inéquation (3.15) devient

$$\begin{aligned} \lambda(x-a) \leqslant f(x) \leqslant \lambda(x-b) &\Leftrightarrow a \leqslant x - \frac{f(x)}{\lambda} \leqslant b \\ &\Leftrightarrow a \leqslant \Phi(x) \leqslant b. \end{aligned}$$

Q. 2 Si $\lambda > 0$, l'inéquation (3.16) devient

$$\begin{aligned} 0 < f'(x) < 2\lambda &\Leftrightarrow 0 < \frac{f'(x)}{\lambda} < 2 \\ &\Leftrightarrow -1 < 1 - \frac{f'(x)}{\lambda} < 1 \\ &\Leftrightarrow -1 < \Phi'(x) < 1. \end{aligned}$$

Si $\lambda < 0$, l'inéquation (3.16) devient

$$\begin{aligned} 2\lambda < f'(x) < 0 &\Leftrightarrow 0 < \frac{f'(x)}{\lambda} < 2 \\ &\Leftrightarrow -1 < 1 - \frac{f'(x)}{\lambda} < 1 \\ &\Leftrightarrow -1 < \Phi'(x) < 1. \end{aligned}$$

Q. 3 Sous les hypothèses (3.15) et (3.16) on a $\Phi([a,b]) \subset [a,b]$ et $\forall x \in [a,b]$, $|\Phi'(x)| < 1$. Comme f est de classe $\mathcal{C}^1$ sur $[a,b]$, la fonction Φ l'est aussi. La suite (x_k) est définie par $x_{k+1} = \Phi(x_k)$. Ainsi les hypothèses du théorème 3.4 sont vérifiées ce qui assure l'unicité du point fixe ainsi que la convergence de la suite (x_k) vers ce point fixe.

◇

Proposition 3.8: ordre de convergence de la méthode de la corde

{RSNL:Corde02}

Soit $f \in \mathcal{C}^1([a,b])$ tel que $f(b) \neq f(a)$. Si la suite (x_k) définie par la méthode de la corde en (3.12) converge vers $\alpha \in]a,b[$ alors la convergence est au moins d'ordre 1.
De plus, si f est de classe $\mathcal{C}^2$ sur un certain voisinage $\mathcal{V}$ de α et si $f'(\alpha) = \frac{f(b)-f(a)}{b-a}$ alors la convergence est au moins d'ordre 2.

Proof.

- **Order 1 :** On note $\lambda = \frac{b-a}{f(b)-f(a)}$. On a par définition $x_{k+1} = x_k - \lambda f(x_k)$ (ce qui suppose que $f(x_k)$ soit bien définie, i.e. $x_k \in [a,b]$). Comme $\lambda \neq 0$ et f continue, l'hypothèse (x_k) converge vers α entraine que $f(\alpha) = 0$.
  Pour définir l'ordre de convergence, on suppose de plus que $\forall k \in \mathbb{N}$, $x_k \neq \alpha$. On peut alors appliquer la formule de Taylor-Lagrange : il existe ξ_k compris entre x_k et α tel que

$$f(x_k) = \underbrace{f(\alpha)}_{=0} + (x_k - \alpha) f'(\xi_k) = (x_k - \alpha) f'(\xi_k).$$

  On a alors en utilisant cette expression dans la définition de la suite x_k

$$x_{k+1} = x_k - \lambda (x_k - \alpha) f'(\xi_k).$$

  En soustrayant α à cette équation on obtient

$$x_{k+1} - \alpha = x_k - \alpha - \lambda (x_k - \alpha) f'(\xi_k) = (x_k - \alpha)(1 - \lambda f'(\xi_k)).$$

  Comme $x_k \neq \alpha$, on a alors

$$\frac{x_{k+1} - \alpha}{x_k - \alpha} = 1 - \lambda f'(\xi_k).$$

  Or x_k converge vers α et ξ_k compris entre x_k et α, ce qui entraine que ξ_k converge vers α. La fonction f' étant continue, on en déduit que $f'(\xi_k)$ converge vers $f'(\alpha)$. Ceci donne donc

$$\lim_{k \to +\infty} \frac{x_{k+1} - \alpha}{x_k - \alpha} = 1 - \lambda f'(\alpha).$$

  La convergence est donc (au moins) d'ordre 1.

- **Order 2 :** La suite étant convergente, il existe $k_0 \in N$ tel que $\forall k \geqslant k_0$, $x_k \in V$. Soit $k \geqslant k_0$, comme $f \in \mathcal{C}^2(\mathcal{V})$, on peut appliquer la formule de Taylor-Lagrange : il existe $\eta_k \in \mathcal{V}$ compris entre x_k et α que

$$f(x_k) = \underbrace{f(\alpha)}_{=0} + (x_k - \alpha) f'(\alpha) + \frac{(x_k - \alpha)^2}{2!} f^{(2)}(\eta_k).$$

  On a alors en utilisant cette expression dans la définition de la suite x_k

$$x_{k+1} = x_k - \lambda \left((x_k - \alpha) f'(\alpha) + \frac{(x_k - \alpha)^2}{2!} f^{(2)}(\eta_k) \right).$$

En soustrayant α à cette équation on obtient

$$x_{k+1}-\alpha = (x_k-\alpha)(\underbrace{1-\lambda f'(\alpha)}_{=0 \text{ par hyp.}}) + \lambda\frac{(x_k-\alpha)^2}{2!}f^{(2)}(\eta_k).$$

Comme $\eta_k \in \mathcal{V}$ converge vers α (car compris entre x_k et α) et $f^{(2)}$ continue sur $\mathcal{V}$, on en déduit

$$\lim_{k\to+\infty}\frac{x_{k+1}-\alpha}{(x_k-\alpha)^2} = \lambda\frac{f^{(2)}(\alpha)}{2}.$$

La convergence est donc (au moins) d'ordre 2.

□

On présente en Algorithme 3.12 l'implémentation usuelle de la méthode de la corde. Une autre version utilisant la fonction PtFixe est donnée en Algorithme 3.13.

{Algo:RSNL:corde0b}

Algorithme 3.12 Méthode de la corde

Données :

f	:	$f : \mathbb{R} \longrightarrow \mathbb{R}$,
$a,\ b$	:	deux réels tels que $f(a) \neq f(b)$,
x_0	:	donnée initiale, $x_0 \in \mathbb{R}$,
tol	:	la tolérence, tol $\in \mathbb{R}^+$,
kmax	:	nombre maximum d'itérations, kmax $\in \mathbb{N}^*$

Résultat :

$\alpha_{\texttt{tol}}$	:	un réel tel que $\lvert f(\alpha_{\texttt{tol}})\rvert \leqslant$ tol

1: **Fonction** $\alpha_{\texttt{tol}} \leftarrow$ Corde(f, a, b, x_0, tol, kmax)
2: $k \leftarrow 0$, $\alpha_{\texttt{tol}} \leftarrow \varnothing$
3: $q \leftarrow \frac{b-a}{f(b)-f(a)}$
4: x $\leftarrow x_0$,
5: err $\leftarrow$ tol $+ 1$
6: **Tantque** err $>$ tol et $k \leqslant$ kmax **faire**
7: $k \leftarrow k+1$
8: xp $\leftarrow$ x
9: x $\leftarrow$ xp $- q * f($xp$)$
10: err $\leftarrow |$x $-$ xp$|$
11: **Fin Tantque**
12: **Si** err $\leqslant$ tol **alors** ▷ Convergence
13: $\alpha_{\texttt{tol}} \leftarrow$ x
14: **Fin Si**
15: **Fin Fonction**

Algorithme 3.13 Méthode de la corde

Données :

f	:	$f : \mathbb{R} \longrightarrow \mathbb{R}$,
$a,\ b$	:	deux réels tels que $f(a) \neq f(b)$,
x_0	:	donnée initiale, $x_0 \in \mathbb{R}$,
tol	:	la tolérence, tol $\in \mathbb{R}^+$,
kmax	:	nombre maximum d'itérations, kmax $\in \mathbb{N}^*$

Résultat :

$\alpha_{\texttt{tol}}$	:	un réel tel que $\lvert f(\alpha_{\texttt{tol}})\rvert \leqslant$ tol

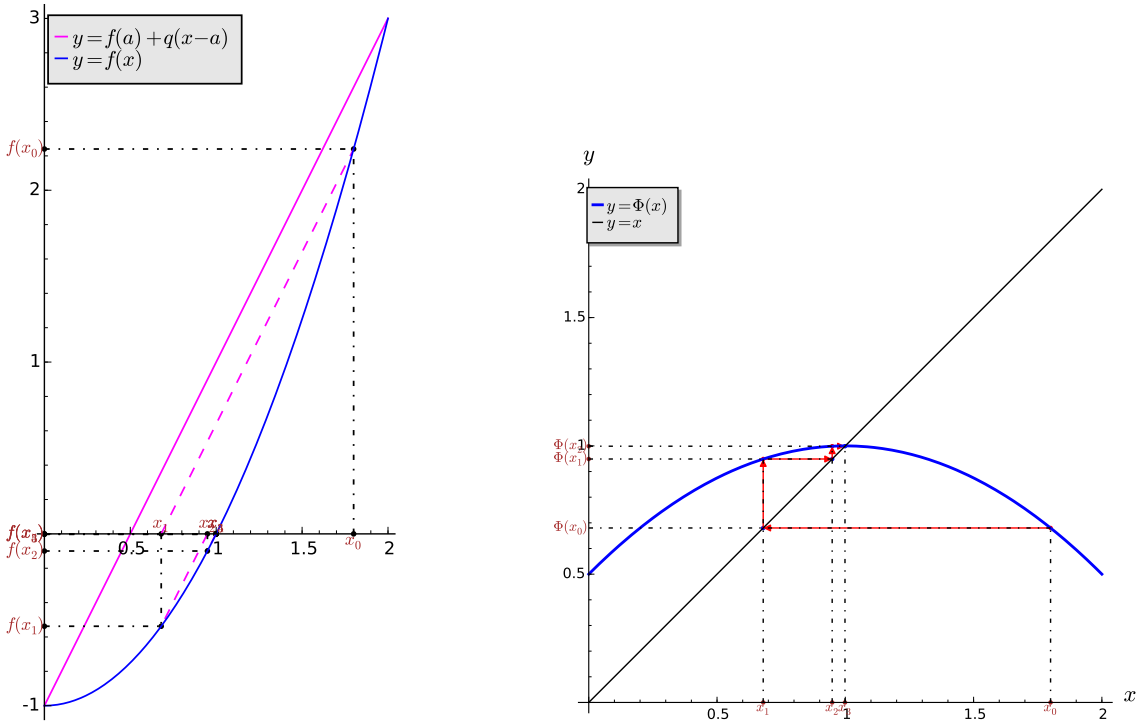
1: **Fonction** $\alpha_{\texttt{tol}} \leftarrow$ Corde(f, a, b, x_0, tol, kmax)
2: $q \leftarrow \frac{b-a}{f(b)-f(a)}$
3: $\Phi \leftarrow x \mapsto x - q * f(x)$
4: $\alpha_{\texttt{tol}} \leftarrow$ PtFixe$(\Phi, x_0$, tol, kmax$)$
5: **Fin Fonction**

Pour illustrer ces résultats de convergence, on va rechercher la racine positive de $f(x) = x^2 - 1$ par la méthode de la corde avec comme données

- exemple 1 : $a = 0.000$, $b = 2.000$, $x_0 = 1.800$,
- exemple 2 : $a = 0.5000$, $b = 1.900$, $x_0 = 1.800$.

On représente en Figure 3.9 les itérations de la méthode de la corde pour l'exemple 1 à partir du graphe de la fonction f (figure de gauche) et à partir du graphe de la fonction Φ (figure de droite). Même chose pour l'exemple 2 avec la Figure 3.10.

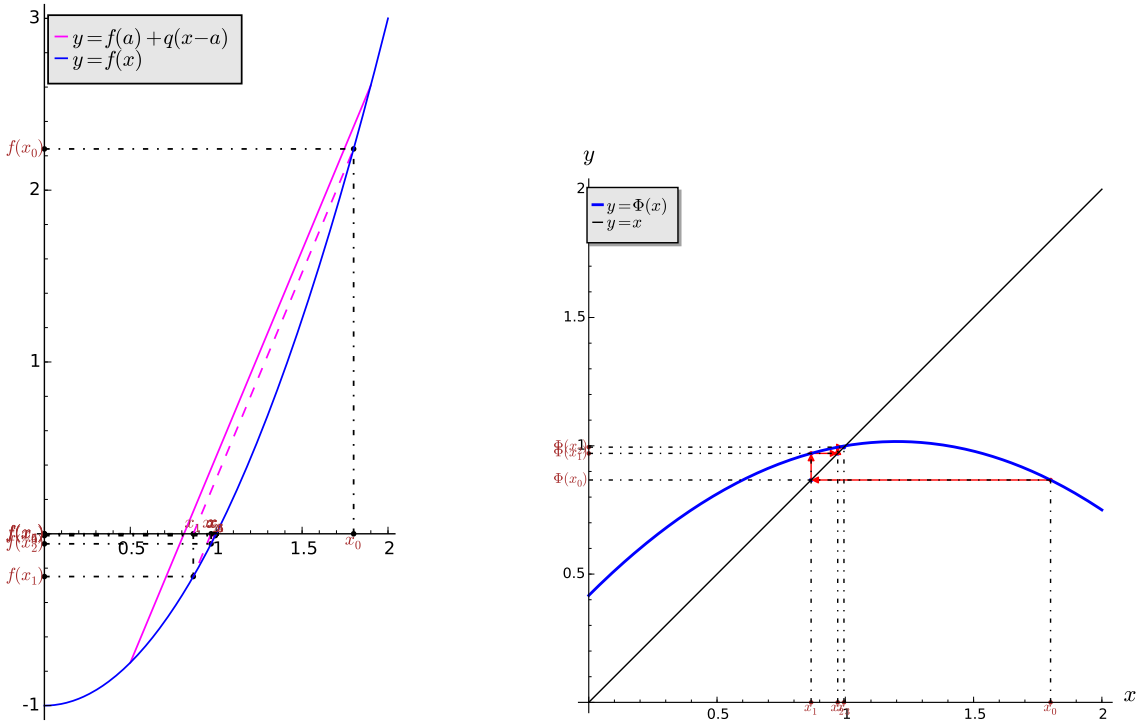
Dans le tableau suivant on donne l'erreur commise par les suites dérivées des exemples 1 et 2 et ceci



(a) représentation usuelle

(b) Représentation point fixe

Figure 3.9: Exemple 1, méthode de la corde, $\alpha = 1$, racine de $f : x \mapsto x^2 - 1$ avec $a = 0.00$, $b = 2.00$, $x_0 = 1.80$,
RSNL:fig:corde:ex01



(a) représentation usuelle

(b) Représentation point fixe

Figure 3.10: Exemple 2, méthode de la corde, $\alpha = 1$, racine de $f : x \mapsto x^2 - 1$ avec $a = 0.50$, $b = 1.90$, $x_0 = 1.80$,
RSNL:fig:corde:ex02

pour quelques itérations.

	exemple 1	exemple 2
k	$\lvert x_k - \alpha\rvert$	$\lvert x_k - \alpha\rvert$
0	8.0000e-01	8.0000e-01
1	3.2000e-01	1.3333e-01
2	5.1200e-02	2.9630e-02
3	1.3107e-03	5.3041e-03
4	8.5899e-07	8.9573e-04
5	3.6893e-13	1.4962e-04
6	0.0000e+00	2.4947e-05
⋮	⋮	⋮
15	0.0000e+00	2.4756e-12

On remarque qu'avec les données de l'exemple 1 la convergence est beaucoup plus rapide. Ceci est

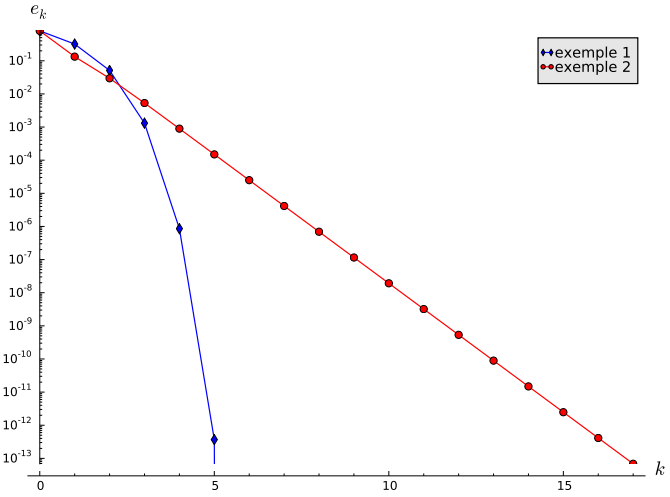


Figure 3.11: Erreurs en fonctions des itérations RSNL:fig:corde:erreur

illustré en Figure 3.11. En effet dans ce cas on a

$$\frac{f(b)-f(a)}{b-a} = 2 \quad \text{et} \quad f'(\alpha) = 2.$$

D'après la proposition 3.8, la convergence est d'ordre 2 pour l'exemple 1. Par contre, on a pour l'exemple 2

$$\frac{f(b)-f(a)}{b-a} = 2.400 \neq f'(\alpha) = 2$$

et donc la convergence est d'ordre 1. On retrouve ces résultats sur la Figure 3.12 ou l'on a représenté en échelle logarithmique $e_{k+1} = |x_{k+1} - \alpha|$ en fonction de $e_k = |x_k - \alpha|$. En effet si une méthode est convergente d'ordre p exactement on aura pour k suffisament grand $e_{k+1} \approx Ce_k^p$.

Ajouter un autre exemple

La méthode de Newton

Soient f une fonction de classe $\mathcal{C}^1$ et x_0 un réel donné. La suite obtenue par la méthode de Newton est donnée par

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \ \forall k \in \mathbb{N}. \tag{3.17}$$

NL:Newton:suite}

Bien évidemment, il faudra s'assurer que $f'(x_k) \neq 0$. On peut voir cette relation comme un cas particulier de la méthode du point fixe. En effet, en prenant $\Phi(x) = x - \frac{f(x)}{f'(x)}$, la suite définie en (3.19) s'écrit $x_{k+1} = \Phi(x_k)$.

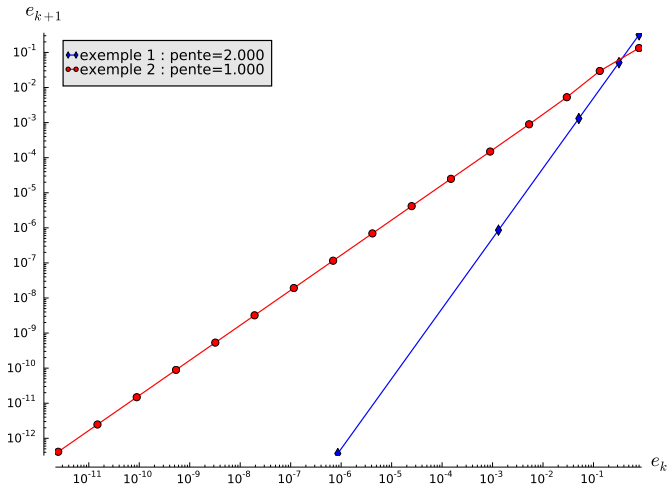


Figure 3.12: Représentation en échelle logarithmique de e_{k+1} en fonction de e_k. Les pentes sont calculées numériquement RSNL:fig:corde:ordre

Proposition 3.9: convergence de la méthode de Newton

SNL:Newton01}

Soit f une fonction de classe $\mathcal{C}^2$ sur un certain voisinage d'une racine simple α de f. Soit x_0 donné dans ce voisinage, la suite $(x_k)_{k\in\mathbb{N}}$ définie par la méthode de Newton

Newton:suite}

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)},\ \forall k \in \mathbb{N}. \tag{3.18}$$

est localement convergente d'ordre 2.

Proof. Comme α est racine simple de f (i.e. $f(\alpha) = 0$ et $f'(\alpha) \neq 0$) et f' continue, il existe un voisinage $\mathcal{V}$ de α tel que pour tout $x \in \mathcal{V}$, $f'(x) \neq 0$. On peut alors définir la fonction Φ sur $\mathcal{V}$ par

$$\forall x \in \mathcal{V},\quad \Phi(x) = x - \frac{f(x)}{f'(x)}.$$

On a alors $x_{k+1} = \Phi(x_k)$. La fonction Φ est de classe $\mathcal{C}^1$ sur $\mathcal{V}$ et

$$\Phi'(x) = 1 - \frac{(f'(x))^2 - f(x)f''(x)}{(f'(x))^2} = \frac{f(x)f''(x)}{(f'(x))^2}.$$

On a donc $\Phi'(\alpha) = 0$ car $f(\alpha) = 0$. D'après le théorème de convergence locale du point fixe (théorème 3.5), on en déduit que la suite (x_k) converge vers α (et que la convergence est au moins d'ordre 1. Pour démontrer qu'elle est d'ordre 2, on ne peut utiliser le théorème 3.6 car Φ n'est pas de classe $\mathcal{C}^2$. Toutefois comme f est de classe $\mathcal{C}^2$, on applique la formule de Taylor-Lagrange aux points x_k, α (en supposant $x_k \neq \alpha$) : il existe ξ_k compris entre x_k et α tel que

$$\underbrace{f(\alpha)}_{=0} = f(x_k) + (\alpha - x_k)f'(x_k) + \frac{(\alpha - x_k)^2}{2!}f^{(2)}(\xi_k).$$

Comme $f'(x_k) \neq 0$, l'équation précédente s'écrit aussi

$$\begin{aligned}\alpha &= x_k - \frac{f(x_k)}{f'(x_k)} + (\alpha - x_k)^2\frac{f^{(2)}(\xi_k)}{2f'(x_k)}\\ &= x_{k+1} + (\alpha - x_k)^2\frac{f^{(2)}(\xi_k)}{2f'(x_k)}.\end{aligned}$$

On obtient alors

$$\frac{\alpha - x_{k+1}}{(\alpha - x_k)^2} = \frac{f^{(2)}(\xi_k)}{2f'(x_k)}.$$

La fonction f est de classe $\mathcal{C}^2$ et la suite ξ_k converge vers α (car ξ_k compris entre x_k et α). Ceci entraine par passage à la limite que

$$\lim_{k\to+\infty} \frac{x_{k+1} - \alpha}{(x_k - \alpha)^2} = \frac{f^{(2)}(\alpha)}{2f'(\alpha)}.$$

La convergence est donc d'ordre 2. □

Soit f une fonction de classe $\mathcal{C}^2$ au voisinage de α, racine de f. On suppose que $f'(x) \neq 0$ pour tout $x \in \mathcal{V}$ (i.e. α racine simple de f). Soit $x_0 \in \mathcal{V}$ donné. La suite obtenue par la méthode de Newton est donnée par

{NL:Newton:suite}

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \ \forall k \in \mathbb{N}. \tag{3.19}$$

On peut voir cette relation comme un cas particulier de la méthode du point fixe. En effet, en prenant $\Phi(x) = x - \frac{f(x)}{f'(x)}$, la suite définie en (3.19) s'écrit $x_{k+1} = \Phi(x_k)$ et on note que $\Phi(\alpha) = \alpha$ (i.e. α point fixe de Φ).
De plus f étant de classe $\mathcal{C}^3$ et sa dérivée non nulle sur $\mathcal{V}$, on obtient que Φ est de classe $\mathcal{C}^2$ sur $\mathcal{V}$, et $\forall x \in \mathcal{V}$,

$$\Phi'(x) = 1 - \frac{(f'(x))^2 - f(x)f''(x)}{(f'(x))^2} = \frac{f(x)f''(x)}{(f'(x))^2}$$

et

$$\Phi''(x) = \frac{(f'(x)f''(x) + f(x)f^{(3)}(x))(f'(x))^2 - 2f(x)f'(x)(f''(x))^2}{(f'(x))^4}$$

On a alors

$$\Phi'(\alpha) = 0, \quad \Phi''(\alpha) = \frac{f''(\alpha)}{f'(\alpha)}.$$

D'après la proposition 3.6, si $f''(\alpha) \neq 0$ alors la **méthode de Newton est d'ordre** 2.
Faire méthode de Newton modifiés dans le cas de racine de multiplicité $m > 1$.???

Exercice 3.2.3

{RSNL:exo104}

En −1700 av. J.-C., les babyloniens ne connaissaient que les nombres rationnels (fractions) et ils utilisaient le système sexagésimal (base 60). Pour approcher la valeur $\sqrt{2}$, ils utilisaient comme approximation (voir tablette YBC 7289)

$$a = 1 + \frac{24}{60} + \frac{51}{60^2} + \frac{10}{60^3} = \frac{30547}{21600}$$

L'erreur commise est $|a - \sqrt{2}| \approx 5.994e - 7$.

Q. 1 *Comment feriez-vous pour trouver* ***à la main*** *une méthode permettant de trouver des nombres rationnels approchant* $\sqrt{2}$.

Q. 2 *Généraliser la méthode pour trouver une approximation rationnelle de* $\sqrt{a}$ *où* a *est un réel positif.*

Q. 3 *Généraliser la méthode pour trouver une approximation rationnelle de* $\sqrt[n]{a}$ *où* a *est un réel positif et* $n \in \mathbb{N}^*$.

Correction Exercice 3.2.3

Q. 1 Il suffit de voir que $\sqrt{2}$ est la racine positive de $f(x) = x^2 - 2$ et d'appliquer la méthode de Newton par exemple. La suite des itérés de Newton s'écrit alors

$$x_{k+1} = x_k - \frac{x_k^2 - 2}{2x_k} = \frac{x_k^2 + 2}{2x_k}$$

Avec $x_0 = 1$, on obtient

k	x_k	$\lvert\sqrt{2} - x_k\rvert$
1	$\frac{3}{2}$	8.57864e-02
2	$\frac{17}{12}$	2.45310e-03
3	$\frac{577}{408}$	2.12390e-06

Avec $x_0 = \frac{3}{2}$, on obtient

k	x_k	$\lvert\sqrt{2} - x_k\rvert$
1	$\frac{17}{12}$	2.45310e-03
2	$\frac{577}{408}$	2.12390e-06
3	$\frac{665857}{470832}$	1.59472e-12

Q. 2 Il suffit de voir que $\sqrt{a}$ est la racine positive de $f(x) = x^2 - a$ et d'appliquer la méthode de Newton par exemple. La suite des itérés de Newton s'écrit alors

$$x_{k+1} = x_k - \frac{x_k^2 - a}{2x_k} = \frac{x_k^2 + a}{2x_k}$$

Avec $a = 3$ et $x_0 = 1$, on obtient

k	x_k	$\lvert\sqrt{3} - x_k\rvert$
1	2	2.67949e-01
2	$\frac{7}{4}$	1.79492e-02
3	$\frac{97}{56}$	9.20496e-05

Q. 3 Il suffit de voir que $\sqrt[n]{a}$ est la racine positive de $f(x) = x^n - a$ et d'appliquer la méthode de Newton par exemple. La suite des itérés de Newton s'écrit alors

$$x_{k+1} = x_k - \frac{x_k^n - a}{2x_k} = \frac{2x_k^2 - x_k^n + a}{2x_k}$$

Avec $a = 3$, $n = 4$ et $x_0 = 1$, on obtient

k	x_k	$\lvert\sqrt[4]{3} - x_k\rvert$
1	$\frac{3}{2}$	1.83926e-01
2	$\frac{97}{72}$	3.11482e-02
3	$\frac{115403137}{87616608}$	1.06368e-03
4	$\frac{2362972972710088378167380851522 57}{1795469431997009848644834162648 32}$	1.28780e-06

◇

On présente en Algorithme 3.14 l'implémentation standard de la méthode de Newton. Une autre version utilisant la fonction PtFixe est donnée en Algorithme 3.15.

Algorithme 3.14 Méthode de Newton {Algo:RSNL:Newton01}

Données :

f	:	$f : \mathbb{R} \longrightarrow \mathbb{R}$,
df	:	la dérivée de f,
x_0	:	donnée initiale, $x_0 \in \mathbb{R}$,
tol	:	la tolérence, tol $\in \mathbb{R}^+$,
kmax	:	nombre maximum d'itérations, kmax $\in \mathbb{N}^*$

Résultat :

$\alpha_{\texttt{tol}}$	:	un réel tel que

1: **Fonction** $\alpha_{\texttt{tol}} \leftarrow$ Newton(f, df, x_0, tol, kmax)
2: $k \leftarrow 0$, $\alpha_{\texttt{tol}} \leftarrow \varnothing$
3: x $\leftarrow x_0$,
4: err $\leftarrow$ tol $+ 1$
5: **Tantque** err $>$ tol et $k \leqslant$ kmax **faire**
6: $k \leftarrow k + 1$
7: xp $\leftarrow$ x
8: x $\leftarrow$ xp $- f($xp$)/$df(xp) ▷ df(xp) $\neq 0$
9: err $\leftarrow |$x $-$ xp$|$
10: **Fin Tantque**
11: **Si** err $\leqslant$ tol **alors** ▷ Convergence
12: $\alpha_{\texttt{tol}} \leftarrow$ x
13: **Fin Si**
14: **Fin Fonction**

Algorithme 3.15 Méthode de Newton {Algo:RSNL:Ne

Données :

f	:	$f : \mathbb{R} \longrightarrow \mathbb{R}$,
df	:	la dérivée de f,
x_0	:	donnée initiale, $x_0 \in \mathbb{R}$,
tol	:	la tolérence, tol $\in \mathbb{R}^+$,
kmax	:	nombre maximum d'itérations, kmax $\in \mathbb{N}^*$

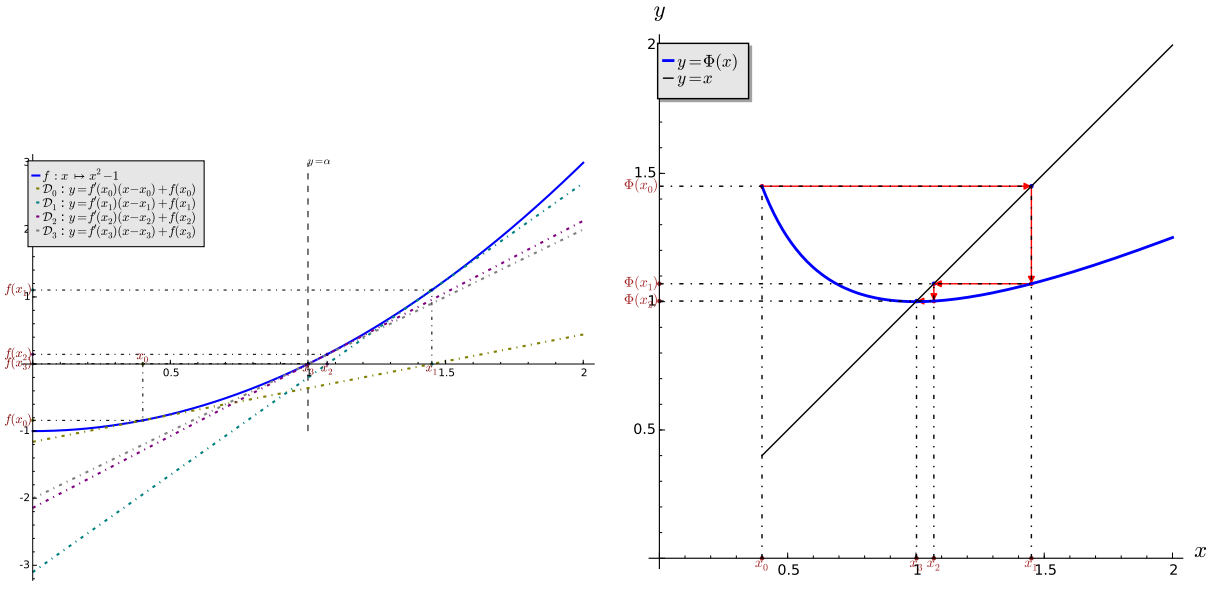
Résultat :

$\alpha_{\texttt{tol}}$	:	un réel tel que

1: **Fonction** $\alpha_{\texttt{tol}} \leftarrow$ Newton(f, df, x_0, tol, kmax)
2: $\Phi \leftarrow x \mapsto x - f(x)/df(x)$
3: $\alpha_{\texttt{tol}} \leftarrow$ PtFixe(Φ, x_0, tol, kmax)
4: **Fin Fonction**

Comme premier exemple, on prend $f(x) = x^2 - 1$ avec $x_0 = 0.40$ et pour le second $f(x) = x^2 \cos(x)$ avec $x_0 = 2.00$.

On représente en Figures 3.13 and 3.14, respectivement pour les exemples 1 et 2, les itérations de la méthode de Newton à partir du graphe de la fonction f (figure de gauche) et à partir du graphe de la fonction Φ (figure de droite).

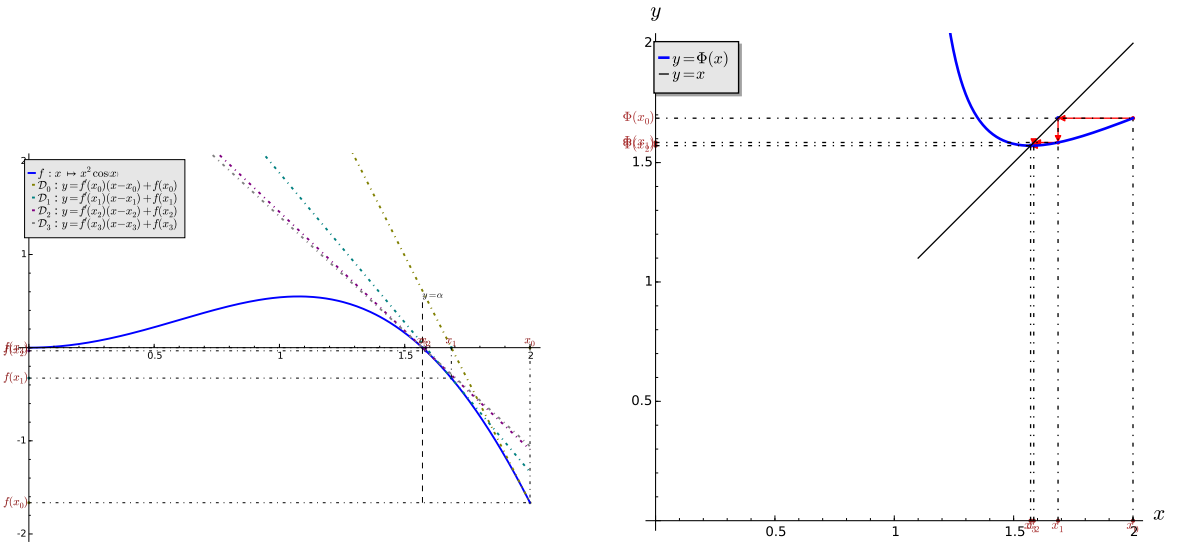


(a) représentation usuelle

(b) Représentation point fixe, $\Phi : x \mapsto x - \frac{x^2-1}{2x}$

Figure 3.13: Exemple 1, méthode de Newton, $\alpha = 1$, racine de $f : x \mapsto x^2 - 1$ avec $x_0 = 0.40$, RSNL:fig:newton:

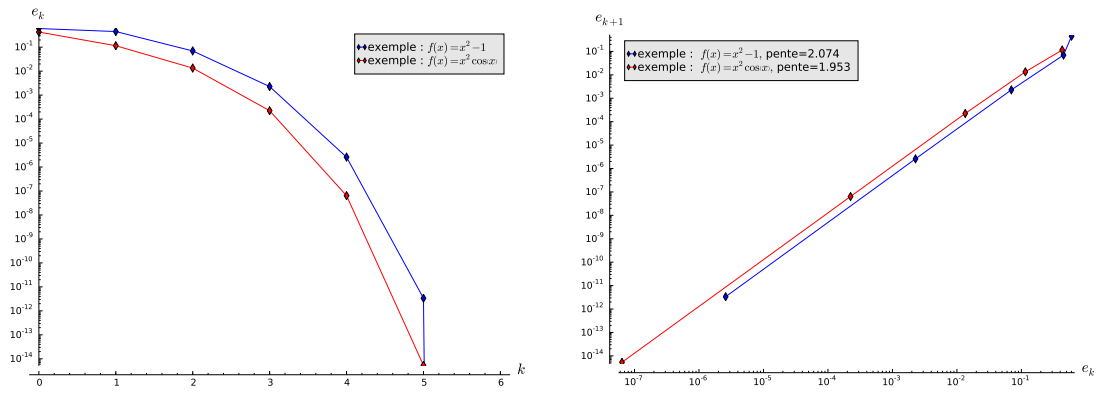
On illustre la convergence et l'ordre de convergence respectivement en Figures 3.15a et 3.15b. Pour cette dernière, on a représenté en échelle logarithmique $e_{k+1} = |x_{k+1} - \alpha|$ en fonction de $e_k = |x_k - \alpha|$. En effet si une méthode est convergente d'ordre p exactement on aura pour k suffisament grand $e_{k+1} \approx C e_k^p$.



(a) représentation usuelle

(b) Représentation point fixe, $\Phi : x \mapsto \frac{x^2\cos(x)}{x^2\sin(x)-2\,x\cos(x)} + x$

Figure 3.14: Exemple 2, méthode de Newton, $\alpha = \frac{1}{2}\pi$, racine de $f : x \mapsto x^2\cos(x)$ avec $x_0 = 0.40$, RSNL:fig:newton:ex2



(a) Représentation de la convergence, e_k en fonction de k {sub@RSNL:Fig:Newton:ordre}

(b) Représentation de l'ordre de convergence en échelle logarithmique, e_{k+1} en fonction de e_k . Ordre théorique 2

n:convergence}

Figure 3.15: Méthode de Newton, convergence et ordre RSNL:Fig:Newton:02

3.2.5 La méthode de la sécante

Cette méthode est une alternative à la méthode de Newton lorsque l'on ne connait pas la dérivée de la fonction f. A l'itération k, de la méthode de Newton, on approche $f'(x_k)$ par $\frac{f(x_k)-f(x_{k-1})}{x_k-x_{k-1}}$. En effet, d'après la formule de Taylor-Lagrange, il existe ξ_k compris entre x_{k-1} et x_k vérifiant

$$f(x_{k-1}) = f(x_k) + (x_{k-1}-x_k)f'(x_k) + \frac{(x_{k-1}-x_k)^2}{2!}f^{(2)}(\xi_k)$$

ce qui donne

$$f'(x_k) = \frac{f(x_k)-f(x_{k-1})}{x_{k-1}-x_k}) + \frac{x_{k-1}-x_k}{2!}f^{(2)}(\xi_k).$$

Si la suite est convergente, on a $\frac{x_{k-1}-x_k}{2!}f^{(2)}(\xi_k) \to 0$, ce qui justifie l'approximation précédente. On a alors la méthode de la sécante donnée en (3.20). Cette méthode est une **méthode à deux pas** : le calcul de x_{k+1} nécessite de connaitre x_k et x_{k-1}. Il faut donc deux valeurs d'initialisations x_{-1} et x_0 pour définir la suite (x_k).

Proposition 3.10: Convergence méthode de la sécante (Admis)

{RSNL:prop:secante01}

Soit f une fonction de classe $\mathcal{C}^2$ sur un certain voisinage d'une racine simple α de f. Soient x_{-1} et x_0 donnés dans ce voisinage tels que $f(x_{-1}) \neq f(x_0)$, la suite $(x_k)_{k\in\mathbb{N}}$ définie par la méthode de la sécante

{RSNL:secante:suite}

$$x_{k+1} = x_k - \frac{x_k - x_{k-1}}{f(x_k)-f(x_{k-1})}f(x_k),\ \forall k \in \mathbb{N}. \tag{3.20}$$

est localement convergente d'ordre $\frac{1+\sqrt{5}}{2} \approx 1.618$.

Comme premier exemple, on recherche une racine de x^2-1 avec $x_{-1} = 0.000$ et $x_0 = 2.000$. Une représentation graphique des premières itérés de la suite est donnée en Figure 3.16. Sur cette figure, les droites $\mathcal{D}_k$ sont celles passant par les points $(x_{k-1}, f(x_{k-1}))$ et $(x_k, f(x_k))$. Pour deuxième exemple, on

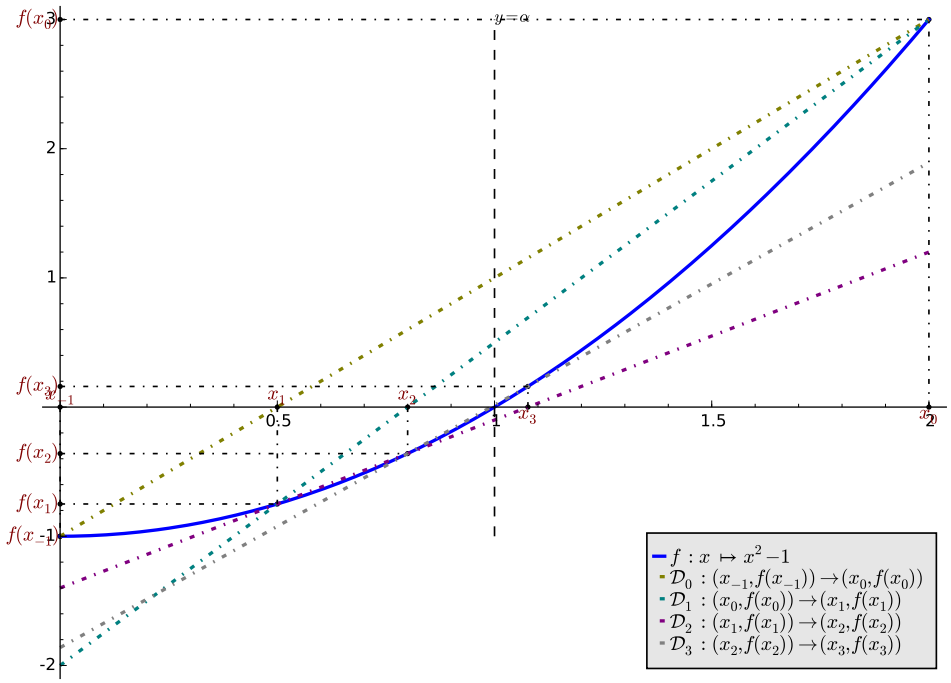


Figure 3.16: Méthode de la sécante pour $f(x) = x^2-1$, $x_{-1} = 0.000$ et $x_0 = 2.000$ {RSNL:Fig:secante01}

recherche une racine de $x^2\cos(x)$ avec $x_{-1} = 1.000$ et $x_0 = 3.000$. Une représentation graphique des premières itérés de la suite est donnée en Figure 3.17.

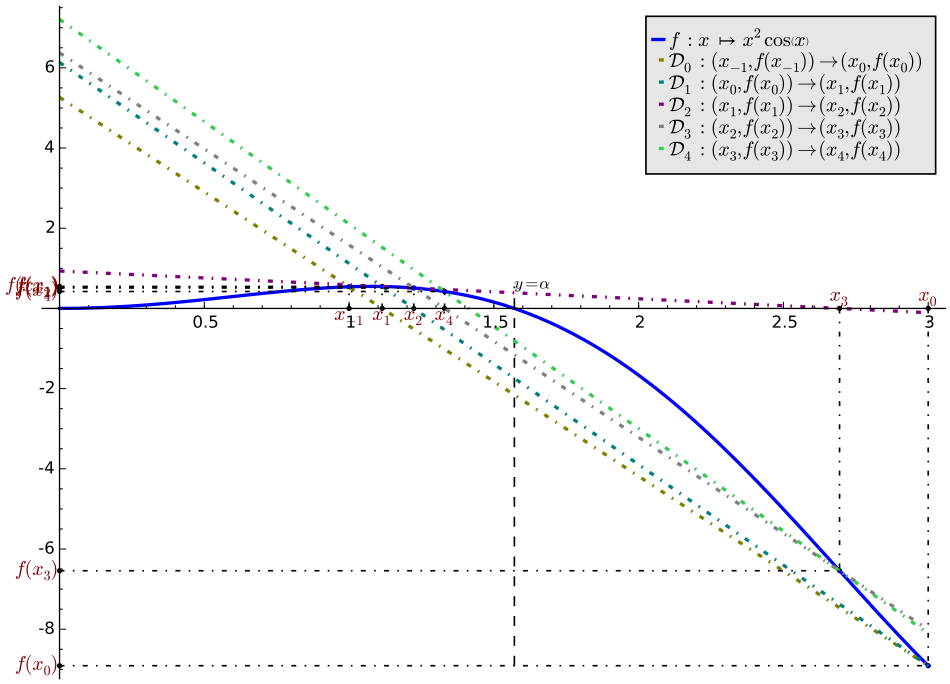
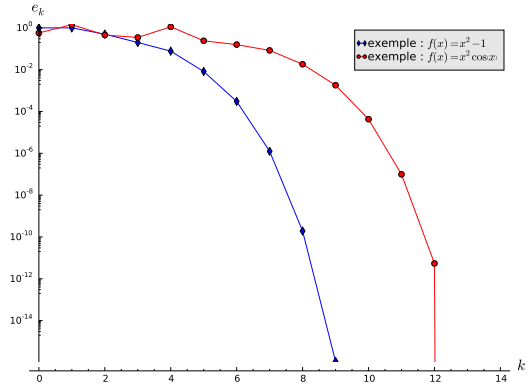
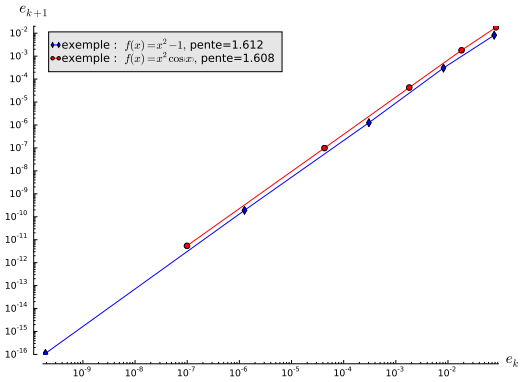


Figure 3.17: Méthode de la sécante pour $f(x) = x^2\cos(x)$, $x_{-1} = 1.000$ et $x_0 = 3.000$ RSNL:Fig:secante02

On illustre la convergence et l'ordre de convergence respectivement en Figures 3.18a et 3.18b. Pour cette dernière, on a représenté en échelle logarithmique $e_{k+1} = |x_{k+1} - \alpha|$ en fonction de $e_k = |x_k - \alpha|$. En effet si une méthode est convergente d'ordre p exactement on aura pour k suffisament grand $e_{k+1} \approx Ce_k^p$.



(a) Représentation de la convergence, e_k en fonction de k



(b) Représentation de l'ordre de convergence en échelle logarithmique e_{k+1} en fonction de e_k . Ordre théorique $\frac{1+\sqrt{5}}{2} \approx 1.618$ {RSNL:Fig:Secante:Ordre} {RSNL:Fig:Secante:02}

Figure 3.18: Méthode de la sécante, convergence et ordre RSNL:Fig:Secante:02

3.2.6 Méthode Regula-Falsi ou fausse position

Cette méthode diffère de la méthode de dichotomie par le choix de x_k à chaque itération. Pour la méthode de dichotomie on a pris $x_k = \frac{a_k+b_k}{2}$ point milieux du segment $[a_k, b_k]$. Pour la méthode Regula-Falsi, on prend pour x_k l'intersection de la droite passant par les points $(a_k, f(a_k))$ et $(b_k, f(b_k))$ avec l'axe des abscisses. Si $f(a_k)f(b_k) < 0$ cela nous assure que $x_k \in]a_k, b_k[$.
L'équation de la droite est donnée par

$$y = cx + d, \text{ avec } c = \frac{f(b_k) - f(a_k)}{b_k - a_k} \text{ et } d = \frac{b_k f(a_k) - a_k f(b_k)}{b_k - a_k}.$$

On a alors

$$x_k = -d/c = \frac{a_k f(b_k) - b_k f(a_k)}{f(b_k) - f(a_k)}.$$

En résumé, on définit les trois suites $(a_k)_{k\in\mathbb{N}}$, $(b_k)_{k\in\mathbb{N}}$ et $(x_k)_{k\in\mathbb{N}}$ par

- $a_0 = a$, $b_0 = b$ et $x_0 = \frac{a_0 f(b_0) - b_0 f(a_0)}{f(b_0) - f(a_0)}$,
- $\forall k \in \mathbb{N}$,

$$\begin{cases} a_{k+1} = b_{k+1} = x_{k+1} = x_k, & \text{si } f(x_k) = 0, \\ a_{k+1} = x_k,\ b_{k+1} = b_k, & \text{si } f(b_k)f(x_k) < 0, \\ a_{k+1} = a_k,\ b_{k+1} = x_k, & \text{si } f(a_k)f(x_k) < 0, \\ x_{k+1} = \frac{a_{k+1} f(b_{k+1}) - b_{k+1} f(a_{k+1})}{f(b_{k+1}) - f(a_{k+1})}, & \text{si } f(x_k) \neq 0. \end{cases}$$

Exercice 3.2.4

{RSNL:exo105}

On suppose que la fonction f est continue sur $[a,b]$, vérifie $f(a)f(b) < 0$ et qu'il existe un unique $\xi \in]a,b[$ tel que $f(x) = 0$.

{exo02-Q1}

Q. 1 *Montrer que*

$$a \leqslant \frac{af(b) - bf(a)}{f(b) - f(a)} \leqslant b.$$

{exo02-Q2}

Q. 2 *Montrer que* $a_0 \leqslant a_1 \leqslant \ldots \leqslant a_k \leqslant x_k \leqslant b_k \leqslant \ldots \leqslant b_1 \leqslant b_0$ *pour tout* $k \in \mathbb{N}$. *et que si* $f(x_k) \neq 0$ *alors* $f(a_k)f(b_k) < 0$.

{exo02-Q3}

Q. 3 *En déduire la convergence de la suite* (x_k) *vers* ξ.

Correction Exercice 3.2.4

Q. 1 On pose $x = \frac{af(b) - bf(a)}{f(b) - f(a)}$ qui est bien défini car $f(a) \neq f(b)$. En effet si $f(a) = f(b)$ alors $f(a)f(b) = f(a)^2 \geqslant 0$ qui est en contradiction avec l'hypothèse $f(a)f(b) < 0$. On a

$$x - a = \frac{af(b) - bf(a) - a(f(b) - f(a))}{f(b) - f(a)} = (b - a)\frac{f(a)}{f(a) - f(b)}$$

Comme $f(a)f(b) < 0$, $f(a) - f(b)$ est du même signe que $f(a)$ et alors $\frac{f(a)}{f(a)-f(b)} \geqslant 0$. De plus $b - a > 0$ et donc on a $x - a \geqslant 0$.
De la même manière, on a

$$x - b = \frac{af(b) - bf(a) - b(f(b) - f(a))}{f(b) - f(a)} = (a - b)\frac{f(b)}{f(b) - f(a)}$$

Comme $f(a)f(b) < 0$, $f(b) - f(a)$ est du même signe que $f(b)$ et alors $\frac{f(b)}{f(b)-f(a)} \geqslant 0$. De plus $a - b < 0$ et donc on a $x - b \leqslant 0$.

Q. 2 On va démontrer par récurrence la validité de la proposition $(\mathcal{P}_k)$ suivante $\forall k \in \mathbb{N}$:

$$(\mathcal{P}_k) \quad \begin{cases} \text{(i)} & f(a_k)f(b_k) < 0 \text{ si } f(x_k) \neq 0 \\ \text{(ii)} & a_0 \leqslant a_1 \leqslant \ldots \leqslant a_k \leqslant x_k \leqslant b_k \leqslant \ldots \leqslant b_1 \leqslant b_0, \end{cases}$$

Initialisation On vérifie la proposition $(\mathcal{P}_k)$ pour $k = 0$. On a $f(a)f(b) < 0$ donc $(\mathcal{P}_0) - (i)$ est vérifiée.
D'après **Q. 1**, on obtient $a_0 \leqslant x_0 \leqslant b_0$. La proposition $(\mathcal{P}_0)$ est donc vérifiée.

Hérédité Soit $k \geqslant 1$. On suppose la proposition $(\mathcal{P}_k)$ est vraie. Montrons que $(\mathcal{P}_{k+1})$ est vérifiée.
Si $f(x_k) = 0$ alors $a_{k+1} = b_{k+1} = x_{k+1} = x_k$ et la proposition $(\mathcal{P}_{k+1})$ est vérifiée.
On suppose maintenant que $f(x_k) \neq 0$.
De $(\mathcal{P}_k) - (i)$ on a $f(a_k) \neq 0$ et $f(b_k) \neq 0$. De plus $f(a_k) \neq f(b_k)$ car sinon $f(a_k)f(b_k) = f(a_k)^2 \geqslant 0$ ce qui est en contradiction avec $(\mathcal{P}_k) - (i)$. Par hypothèse $(\mathcal{P}_k)$, on a $a_k \leqslant x_k \leqslant b_k$ et $f(a_k)f(b_k) < 0$.
Par continuité de f, on a alors soit $f(b_k)f(x_k) < 0$ (et donc $f(a_k)f(x_k) > 0$) soit $f(b_k)f(x_k) > 0$ (et donc $f(a_k)f(x_k) < 0$).

- Si $f(b_k)f(x_k) < 0$ on a $a_{k+1} = x_k$ et $b_{k+1} = b_k$. Comme $f(a_k) \neq f(b_k)$, x_{k+1} est bien défini. D'après **Q. 1** en prenant $[a_{k+1}, b_{k+1}]$ comme intervalle, et sachant que $f(a_{k+1})f(b_{k+1}) = f(x_k)f(b_k) < 0$ on obtient
$$a_{k+1} \leqslant x_{k+1} \leqslant b_{k+1}.$$
De plus par hypothèse $a_k \leqslant x_k = a_{k+1}$ et donc $a_k \leqslant a_{k+1}$ et $b_{k+1} \leqslant b_k$. La proposition $(\mathcal{P}_{k+1})$ est donc vérifiée.
- Si $f(a_k)f(x_k) < 0$ on a $a_{k+1} = a_k$ et $b_{k+1} = x_k$. Comme $f(a_k) \neq f(b_k)$, x_{k+1} est bien défini. D'après **Q. 1** en prenant $[a_{k+1}, b_{k+1}]$ comme intervalle, et sachant que $f(a_{k+1})f(b_{k+1}) = f(a_k)f(x_k) < 0$ on obtient
$$a_{k+1} \leqslant x_{k+1} \leqslant b_{k+1}.$$
La proposition $(\mathcal{P}_{k+1})$ est donc vérifiée.

Q. 3 Supposons qu'il existe $s \in \mathbb{N}$ tel que $f(x_s) = 0$. Alors, pour tout $i \leqslant 1$, on a $a_{s+i} = b_{s+i} = x_{s+i} = x_s$. Les trois suites convergent donc vers x_s. D'après la question précédente, $x_s \in [a, b]$. Comme $f(a) \neq 0$ et $f(b) \neq 0$, on en déduit $x_s \in]a, b[$. Par hypothèse il existe un unique $\xi \in]a, b[$ tel que $f(\xi) = 0$, on a alors $x_s = \xi$.
Supposons que $\forall k \in \mathbb{N}$, $f(x_k) \neq 0$. D'après **Q. 2**, la suite (a_k) est croissante majorée par b et la suite (b_k) est décroissante minorée par a. Elles sont donc convergentes et l'on note respectivement l et L les limites de (a_k) et (b_k). Comme $a \leqslant a_k \leqslant b_k \leqslant b$, on a $a \leqslant l \leqslant L \leqslant b$.

- Supposons $f(l) = f(L)$. On a $f(a_k)f(b_k) < 0$. Comme f est continue, à la limite on obtient $f(l)f(L) = f(l)^2 = f(L)^2 \leqslant 0$ et donc $f(l) = f(L) = 0$. On a necessairement l et L dans $]a, b[$ car $f(a)f(b) < 0$ et donc $f(a)$ et $f(b)$ non nuls. Par unicité du zéro de f dans $]a, b[$ on obtient $l = L = \xi$. Comme $a_k \leqslant x_k \leqslant b_k$, on en déduit que la suite (x_k) converge aussi vers ξ.
- Supposons $f(l) \neq f(L)$. Par continuité de la fonction f la suite (x_k) converge alors vers $M = \frac{lf(L) - Lf(l)}{f(L) - f(l)}$. Comme $a_k \leqslant x_k \leqslant b_k$ on a aussi
$$l \leqslant M = \frac{lf(L) - Lf(l)}{f(L) - f(l)} \leqslant L. \tag{3.21}$$ {exo02:eq01}
De plus ayant $f(x_k) \neq 0$ $\forall k \in \mathbb{N}$, on a $f(a_k)f(b_k) < 0$ $\forall k \in \mathbb{N}$. En passant à la limite et par continuité de f on obtient $f(l)f(L) \leqslant 0$.
Montrons que $f(l) = 0$ ou $f(L) = 0$.
  - Si $f(l) < f(L)$, alors de (3.21) on obtient
$$l(f(L) - f(l)) \leqslant lf(L) - Lf(l) \leqslant L(f(L) - f(l))$$
ce qui donne $lf(l) \geqslant Lf(l)$ et $lf(L) \leqslant Lf(L)$ i.e. $(l - L)f(l) \geqslant 0$ et $(L - l)f(L) \leqslant 0$. Comme $L - l \geqslant 0$, on en déduit $f(l) \leqslant 0$ et $f(L) \leqslant 0$. Or $f(l)f(L) \leqslant 0$, ce qui donne $f(l) = 0$ ou $f(L) = 0$.
  - Si $f(l) > f(L)$, alors de (3.21) on obtient
$$l(f(L) - f(l)) \geqslant lf(L) - Lf(l) \geqslant L(f(L) - f(l))$$
ce qui donne $lf(l) \leqslant Lf(l)$ et $lf(L) \geqslant Lf(L)$ i.e. $(L - l)f(l) \geqslant 0$ et $(L - l)f(L) \leqslant 0$. Comme $L - l \geqslant 0$, on en déduit $f(l) \geqslant 0$ et $f(L) \geqslant 0$. Or $f(l)f(L) \leqslant 0$, ce qui donne $f(l) = 0$ ou $f(L) = 0$.

  On a donc démontré que si $f(l) \neq f(L)$, alors $f(l) = 0$ ou $f(L) = 0$ et donc
  - si $f(l) = 0$ alors $M = \frac{lf(L) - Lf(l)}{f(L) - f(l)} = l$ et donc $f(M) = 0$.
  - si $f(L) = 0$ alors $M = \frac{lf(L) - Lf(l)}{f(L) - f(l)} = L$ et donc $f(M) = 0$.

  Puisque l et L appartiennent à $]a, b[$, on a $M \in]a, b[$. Par unicité du zéro de f sur $]a, b[$ on en déduit que $M = \xi$.

◇

On vient de démontrer le théorème suivant

Théorème 3.11 {RSNL:Theo:Re

Soit $f : [a,b] \subset \mathbb{R} \longrightarrow \mathbb{R}$ une fonction continue vérifiant $f(a)f(b) < 0$ et admettant $\alpha \in]a,b[$ comme **unique** solution de $f(x) = 0$. Alors la suite $(x_k)_{k\in\mathbb{N}}$ définie par la **méthode Regula-Falsi** converge vers α

On a aussi la proposition suivante

Proposition 3.12: ordre de la méthode de Regula-Falsi o:RegulaFalsiOrdre}

Soit $f : [a,b] \subset \mathbb{R} \longrightarrow \mathbb{R}$ une fonction continue vérifiant $f(a)f(b) < 0$ et admettant $\alpha \in]a,b[$ comme **unique** solution de $f(x) = 0$. Si f est deux fois dérivables sur $[a,b]$ et si f'' est monotone sur $]a,b[$ alors il existe $C \in \mathbb{R}$ tel que

$$\lim_{k\to+\infty} \frac{|x_{k+1} - \alpha|}{|x_k - \alpha|} \leqslant C \tag{3.22}$$

La méthode de Regula-Falsi est alors à convergence linéaire (d'ordre 1).

3.3 Résolution de systèmes non linéaires {RSNL:sec:dD}

Nous allons (très rapidement) introduire les premières notions permettant la résolution de systèmes d'équations non linéaires. Par exemple, dans $\mathbb{R}^2$ nous allons regarder le problème suivant avec c une constante réelle

$$\begin{cases} f_1(x_1,x_2) = -x_1^3 + x_2 - \frac{1}{2} & = 0 \\ f_2(x_1,x_2) = \frac{1}{25}\left(10\,x_2 + 1\right)^2 + c - x_1 & = 0. \end{cases} \tag{3.23}$$ SNL:eq:system01}

En Figures 3.19 et 3.21, on représente pour différentes valeurs de c les courbes $f_1(x_1,x_2) = 0$ et $f_2(x_1,x_2) = 0$: les intersections de ces deux courbes sont les solutions du problème (3.23). Comme on le voit graphiquemment, il peut y avoir, suivant les valeurs de c, 0, 1, 2 ou 4 solutions.

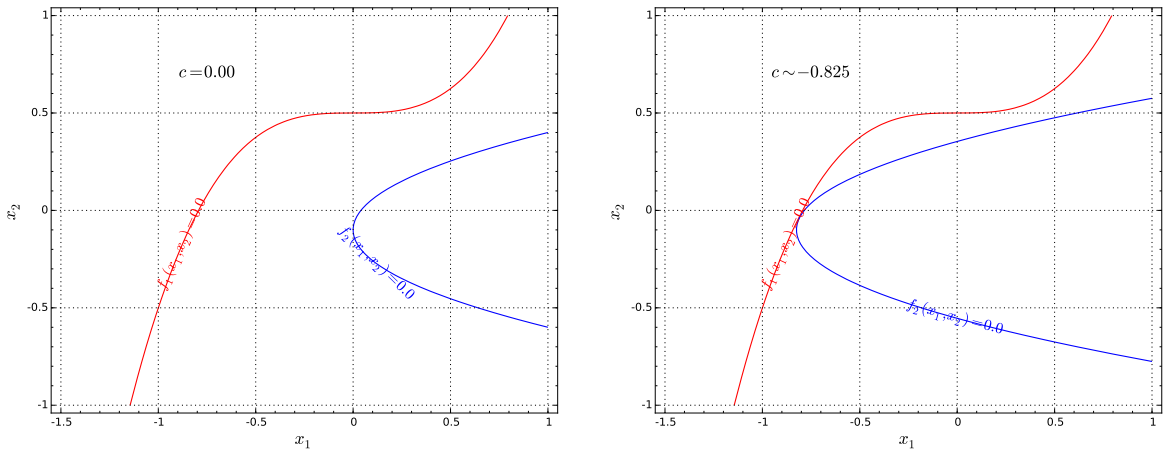


Figure 3.19: Résolution graphique de 3.23 avec $c = 0.00$ (gauche) et $c \sim -0.825$ (droite) RSNL:fig:system:01

De manière plus générale, soient $U \subset \mathbb{R}^N$ un ouvert et $\boldsymbol{f}$ une application continue de U dans $\mathbb{R}^N$. Le problème que l'on souhaite résoudre est le suivant

Trouver $\boldsymbol{\alpha} \in U \subset \mathbb{R}^N$ tel que

$$\boldsymbol{f}(\boldsymbol{\alpha}) = 0 \iff \begin{cases} \boldsymbol{f}_1(\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_N) &= 0 \\ \boldsymbol{f}_2(\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_N) &= 0 \\ \vdots & \\ \boldsymbol{f}_N(\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_N) &= 0 \end{cases}$$

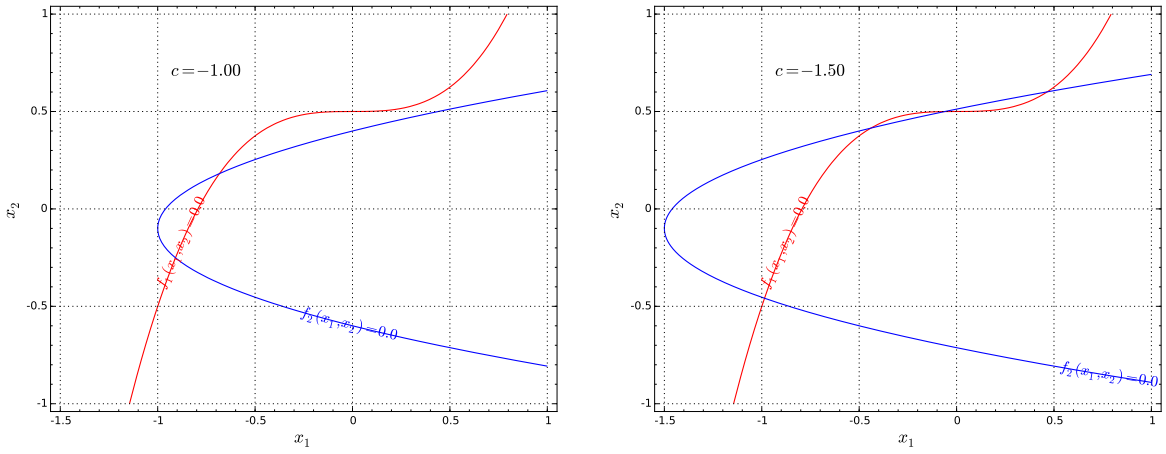


Figure 3.20: Résolution graphique de 3.23 avec $c=-1.00$ (gauche) et $c=-1.50$ (droite) RSNL:fig:system:02

Comme dans le cas scalaire, pour résoudre numériquement ce genre de problème on utilise des suites itératives et plus particulièrement celles basées sur les méthodes de points fixes. En effet, nous allons voir que le théorème du point fixe se généralise très facilement (voir Théorème 3.13).

En définissant, par exemple, la fonction $\boldsymbol{\Phi} \in \mathcal{C}^0(U;\mathbb{R}^N)$ par $\boldsymbol{\Phi}(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{f}(\boldsymbol{x})$, on peut remarquer $\boldsymbol{f}(\boldsymbol{x}) = 0$ est équivalent à $\boldsymbol{\Phi}(\boldsymbol{x}) = \boldsymbol{x}$. On peut donc se ramener à la recherche d'un point fixe (s'il existe) de la fonction $\boldsymbol{\Phi}$.

Trouver $\boldsymbol{\alpha} \in U \subset \mathbb{R}^N$ tel que

$$\boldsymbol{\Phi}(\boldsymbol{\alpha}) = \boldsymbol{\alpha} \iff \begin{cases} \boldsymbol{\Phi}_1(\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_N) &= \boldsymbol{\alpha}_1 \\ \boldsymbol{\Phi}_2(\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_N) &= \boldsymbol{\alpha}_2 \\ \vdots & \\ \boldsymbol{\Phi}_N(\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_N) &= \boldsymbol{\alpha}_N \end{cases}$$

Les suites itératives sont donc de la forme

$$\boldsymbol{x}^{[k+1]} = \boldsymbol{\Phi}(\boldsymbol{x}^{[k]})$$

où $\boldsymbol{\Phi}$ est une fonction à déterminer et $\boldsymbol{x}^{[0]} \in \mathbb{R}^N$. Bien évidemment le choix d'une *bonne* fonction $\boldsymbol{\Phi}$ est primordiale pour espérer avoir convergence.

Ce type de problème peut s'avérer délicat à traiter : comment choisir $\boldsymbol{\Phi}$? $\boldsymbol{x}^{[0]}$? Si l'on converge vers quel point fixe?

3.3.1 Point fixe

Théorème 3.13

{RSNL:Theo:Pt

Soit $\mathcal{B}$ un espace de Banach et $U \subset \mathcal{B}$ un sous-ensemble fermé. On suppose que $\boldsymbol{\Phi} : U \longrightarrow U$ est une application strictement contractante, i.e.

$$\exists L \in]0,1[, \quad \|\boldsymbol{\Phi}(\boldsymbol{x}) - \boldsymbol{\Phi}(\boldsymbol{y})\| \leqslant \|\boldsymbol{x} - \boldsymbol{y}\|, \quad \forall (\boldsymbol{x}, \boldsymbol{y}) \in U \times U. \tag{3.24}$$ {PointFixeBar

Alors

1. $\boldsymbol{\Phi}$ admet un unique point fixe $\boldsymbol{\alpha} \in U$ (i.e. unique solution de $\boldsymbol{x} = \boldsymbol{\Phi}(\boldsymbol{x})$).
2. La suite des itérés $\boldsymbol{x}^{[k+1]} = \boldsymbol{\Phi}(\boldsymbol{x}^{[k]})$ converge vers $\boldsymbol{\alpha}$ pour toute valeur initiale $\boldsymbol{x}^{[0]} \in U$.
3. Pour tout $k \in \mathbb{N}$,

$$\left\|\boldsymbol{\alpha} - \boldsymbol{x}^{[k]}\right\| \leqslant \frac{L^{k-l}}{1-L} \left\|\boldsymbol{x}^{[l+1]} - \boldsymbol{x}^{[l]}\right\|, \quad 0 \leqslant l \leqslant k \tag{3.25}$$ {PointFixeBar

1

Proof. On démontre tout d'abord l'existence d'un point fixe. Pour celà on va démontrer que la suite $\boldsymbol{x}^{[k]}$ est de Cauchy dans U fermé d'un espace de Banach (donc elle converge dans U).
Comme $\boldsymbol{\Phi}$ est contractante, on a pour tout $k \in \mathbb{N}^*$

$$\left\|\boldsymbol{x}^{[k+1]} - \boldsymbol{x}^{[k]}\right\| = \left\|\boldsymbol{\Phi}(\boldsymbol{x}^{[k]}) - \boldsymbol{\Phi}(\boldsymbol{x}^{[k-1]})\right\| \leqslant L \left\|\boldsymbol{x}^{[k]} - \boldsymbol{x}^{[k-1]}\right\|$$

ce qui donne par récurrence pour tout $0 \leqslant j \leqslant k$

nach:proof:eq01}
$$\left\|\boldsymbol{x}^{[k+1]} - \boldsymbol{x}^{[k]}\right\| \leqslant L^j \left\|\boldsymbol{x}^{[k+1-j]} - \boldsymbol{x}^{[k-j]}\right\| \tag{3.26}$$

ou encore pour tout $0 \leqslant l \leqslant k$, $(l = k - j)$

nach:proof:eq02}
$$\left\|\boldsymbol{x}^{[k+1]} - \boldsymbol{x}^{[k]}\right\| \leqslant L^{k-l} \left\|\boldsymbol{x}^{[l+1]} - \boldsymbol{x}^{[l]}\right\| \tag{3.27}$$

On obtient aussi par récurrence

nach:proof:eq03}
$$\forall l \geqslant 0, \quad \left\|\boldsymbol{x}^{[k+1+l]} - \boldsymbol{x}^{[k+l]}\right\| \leqslant L^l \left\|\boldsymbol{x}^{[k+1]} - \boldsymbol{x}^{[k]}\right\|. \tag{3.28}$$

Soit $p \geqslant 1$. On en déduit par application répétée de l'inégalité triangulaire que

$$\begin{aligned}
\left\|\boldsymbol{x}^{[k+p]} - \boldsymbol{x}^{[k]}\right\| &= \left\|(\boldsymbol{x}^{[k+p]} - \boldsymbol{x}^{[k+p-1]}) + (\boldsymbol{x}^{[k+p-1]} - \boldsymbol{x}^{[k+p-2]}) + \ldots + (\boldsymbol{x}^{[k+1]} - \boldsymbol{x}^{[k]})\right\| \\
&= \left\|\sum_{l=0}^{p-1} (\boldsymbol{x}^{[k+l+1]} - \boldsymbol{x}^{[k+l]})\right\| \\
&\leqslant \sum_{l=0}^{p-1} \left\|\boldsymbol{x}^{[k+l+1]} - \boldsymbol{x}^{[k+l]}\right\| \\
&\overset{(3.28)}{\leqslant} \sum_{l=0}^{p-1} L^l \left\|\boldsymbol{x}^{[k+1]} - \boldsymbol{x}^{[k]}\right\| = \frac{1-L^p}{1-L} \left\|\boldsymbol{x}^{[k+1]} - \boldsymbol{x}^{[k]}\right\| \\
&\leqslant \frac{1-L^p}{1-L} L^k \left\|\boldsymbol{x}^{[1]} - \boldsymbol{x}^{[0]}\right\|. \quad \textit{(en utilisant (3.27), avec } l = 0\textit{)}
\end{aligned}$$

Comme $L^k \to 0$ quand $k \to +\infty$, on conclu que $(\boldsymbol{x}^{[k]})$ est une suite de Cauchy. De plus, par construction $\boldsymbol{x}^{[k]} \in U \subset \mathcal{B}$, pour tout $k \in \mathbb{N}$, et $\mathcal{B}$ étant un espace de Banach et U un fermé, la suite $(\boldsymbol{x}^{[k]})$ converge alors vers $\boldsymbol{\alpha} \in U$. Comme $\boldsymbol{\Phi}$ est contractante, elle est donc continue et en passant à la limite dans $\boldsymbol{x}^{[k+1]} = \boldsymbol{\Phi}(\boldsymbol{x}^{[k]})$, on abouti à $\boldsymbol{\alpha} = \boldsymbol{\Phi}(\boldsymbol{\alpha})$, i.e. $\boldsymbol{\alpha}$ est un point fixe de Φ dans U.
L'unicité se déduit immédiatement par la contraction de la fonction $\boldsymbol{\Phi}$. En effet, soit $\boldsymbol{\alpha}_1$ et $\boldsymbol{\alpha}_2$ deux points fixes de $\boldsymbol{\Phi}$, alors

$$\|\boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_2\| = \|\boldsymbol{\Phi}(\boldsymbol{\alpha}_1) - \boldsymbol{\Phi}(\boldsymbol{\alpha}_2)\| \leqslant L \|\boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_2\|$$

2 Or $L < 1$, et donc nécessairement on a $\boldsymbol{\alpha_1 = \alpha_2}$.

Il reste à démontrer l'inégalité (3.25). On a vu que pour $p \geqslant 1$

$$\left\|\boldsymbol{x}^{[k+p]} - \boldsymbol{x}^{[k]}\right\| \leqslant \frac{1-L^p}{1-L}\left\|\boldsymbol{x}^{[k+1]} - \boldsymbol{x}^{[k]}\right\|$$

L'application norme étant continue et $L < 1$, on obtient à la limite quand $p \to +\infty$

$$\left\|\boldsymbol{\alpha} - \boldsymbol{x}^{[k]}\right\| \leqslant \frac{1}{1-L}\left\|\boldsymbol{x}^{[k+1]} - \boldsymbol{x}^{[k]}\right\|$$

On obtient l'inégalité en utilisant (3.27). □

3.3.2 Méthode de Newton

On commence par rappeler un résultat de calcul différentiel. Soit $\boldsymbol{f} : \mathbb{R}^N \longrightarrow \mathbb{R}^N$ une fonction suffisament régulière. On défini la **matrice Jacobienne de** $\boldsymbol{f}$, notée $\mathbb{J}_{\boldsymbol{f}}$, par

$$\mathbb{J}_{\boldsymbol{f}} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_N} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_N} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_N}{\partial x_1} & \frac{\partial f_N}{\partial x_2} & \cdots & \frac{\partial f_N}{\partial x_N} \end{pmatrix}$$

On a alors $\forall \boldsymbol{h} \in \mathbb{R}^N$ à l'ordre 1

$$\boldsymbol{f}(\boldsymbol{x}+\boldsymbol{h}) \approx \boldsymbol{f}(\boldsymbol{x}) + \mathbb{J}_{\boldsymbol{f}}(\boldsymbol{x}).\boldsymbol{h}. \tag{3.29}$$ {RSNL:TaylordN}

Nous voulons trouver $\boldsymbol{\alpha}$ tel que $\boldsymbol{f}(\boldsymbol{\alpha}) = 0$. Si $\boldsymbol{x}^{[k]}$ est proche de $\boldsymbol{\alpha}$, alors en utilisant (3.29) avec $\boldsymbol{x} = \boldsymbol{x}^{[k]}$ et $\boldsymbol{\alpha} = \boldsymbol{x}^{[k]} + \boldsymbol{h}$ (i.e. $\boldsymbol{h} = \boldsymbol{\alpha} - \boldsymbol{x}^{[k]}$) on obtient

$$\boldsymbol{f}(\boldsymbol{\alpha}) \approx \boldsymbol{f}(\boldsymbol{x}^{[k]}) + \mathbb{J}_{\boldsymbol{f}}(\boldsymbol{x}^{[k]}).\boldsymbol{h}$$

Au lieu de résoudre $\boldsymbol{f}(\boldsymbol{x}) = 0$, on résoud le système linéarisé

$$\boldsymbol{f}(\boldsymbol{x}^{[k]}) + \mathbb{J}_{\boldsymbol{f}}(\boldsymbol{x}^{[k]}).\tilde{\boldsymbol{h}} = 0$$

c'est à dire le système linéaire

$$\mathbb{J}_{\boldsymbol{f}}(\boldsymbol{x}^{[k]}).\tilde{\boldsymbol{h}} = -\boldsymbol{f}(\boldsymbol{x}^{[k]}). \tag{3.30}$$ {RSNL:NewtonEd.e

On espère alors que $\tilde{\boldsymbol{h}}$ est une bonne approximation de $\boldsymbol{h}$ au sens ou $\boldsymbol{x}^{[k]}+\tilde{\boldsymbol{h}}$ est une meilleur approximation de $\boldsymbol{\alpha}$ que $\boldsymbol{x}^{[k]}$. On note alors $\boldsymbol{x}^{[k+1]} = \boldsymbol{x}^{[k]} + \tilde{\boldsymbol{h}}$. En posant $\Phi(\boldsymbol{x}) = \boldsymbol{x} - ((\mathbb{J}_{\boldsymbol{f}}(\boldsymbol{x}))^{-1}\boldsymbol{f}(\boldsymbol{x})$ la méthode de Newton s'écrit alors

$$\boldsymbol{x}^{[k+1]} = \Phi(\boldsymbol{x}^{[k]}) = \boldsymbol{x}^{[k]} - \left((\mathbb{J}_{\boldsymbol{f}}(\boldsymbol{x}^{[k]})\right)^{-1}\boldsymbol{f}(\boldsymbol{x}^{[k]}) \tag{3.31}$$

Cette formule est une généralisation de celle vue dans le cas scalaire (voir ??). Il faut noter qu'a chaque itération la matrice Jacobienne est modifiée et qu'il faut calculer le vecteur $-\left((\mathbb{J}_{\boldsymbol{f}}(\boldsymbol{x}^{[k]})\right)^{-1}\boldsymbol{f}(\boldsymbol{x}^{[k]})$. Numeriquement, on ne calcule que très rarement l'inverse d'une matrice car celà est très couteux en temps mais on résoud le système linéaire (3.30) ce qui est bien plus efficace.

On admet dans ce cours le théorème suivant

Théorème 3.14

Soit $\boldsymbol{f} \in \mathcal{C}^3(\mathbb{R}^N;\mathbb{R}^N)$. On suppose que la matrice Jacobienne appliquée en $\boldsymbol{x}$, $\mathbb{J}_{\boldsymbol{f}}(\boldsymbol{x})$ est inversible dans un voisinage de $\boldsymbol{\alpha}$, avec $\boldsymbol{f}(\boldsymbol{\alpha}) = 0$. Alors pour tout $\boldsymbol{x}^{[0]}$ suffisament proche de $\boldsymbol{\alpha}$ la suite définie par

$$\boldsymbol{x}^{[k+1]} = \boldsymbol{x}^{[k]} - \left((\mathbb{J}_{\boldsymbol{f}}(\boldsymbol{x}^{[k]})\right)^{-1}\boldsymbol{f}(\boldsymbol{x}^{[k]})$$

converge vers $\boldsymbol{\alpha}$ et la convergence est d'ordre 2.

On donne ensuite l'algorithme 3.16 permettant de déterminer une approximation d'un point fixe d'une fonction $\boldsymbol{f}$. Dans cet algorithme on suppose donnée la fonction Solve permettant de résoudre un système linéaire.

Algorithme 3.16 Méthode de Newton {Algo:RSNL

Données :

$\boldsymbol{f}$	:	$\boldsymbol{f} : \mathbb{R}^N \longrightarrow \mathbb{R}^N$,
Jf	:	la matrice Jacobienne de f,
x0	:	donnée initiale, $\texttt{x0} \in \mathbb{R}^N$,
tol	:	la tolérence, $\texttt{tol} \in \mathbb{R}^+$,
kmax	:	nombre maximum d'itérations, $\texttt{kmax} \in \mathbb{N}^*$

Résultat :

$\boldsymbol{\alpha}_{\texttt{tol}}$	:	un élément de $\mathbb{R}^N$ proche de $\boldsymbol{\alpha}$.

Fonction $\alpha_{\texttt{tol}} \leftarrow$ Newton($\boldsymbol{f}$, Jf, x0, tol, kmax)
  $k \leftarrow 0$, $\alpha_{\texttt{tol}} \leftarrow \varnothing$
  x $\leftarrow$ x0,
  err $\leftarrow$ tol $+ 1$
  Tantque err $>$ tol et $k \leqslant$ kmax **faire**
    $k \leftarrow k + 1$
    xp $\leftarrow$ x
    h $\leftarrow$ Solve(Jf(xp), $-f$(xp)) ▷ $\boldsymbol{x} \leftarrow$ Solve($\mathbb{A}, \boldsymbol{b}$) : résoud le système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$
    x $\leftarrow$ xp + h
    err $\leftarrow$ Norm(x − xp)
  Fin Tantque
  Si err $\leqslant$ tol **alors** ▷ Convergence
    $\alpha_{\texttt{tol}} \leftarrow$ x
  Fin Si
Fin Fonction

Remarque 3.15 Si l'on ne connait pas explicitement la Jacobienne de f, il est possible de calculer une approximation de celle-ci en utilisant des formules de dérivation numérique.

3.3.3 Exemples

Exemple modèle

Comme premier exemple, nous reprenons le système 3.23 avec $c = -1.5$

NS:eq:system01a}
$$\begin{cases} f_1(x_1, x_2) = -x_1^3 + x_2 - \frac{1}{2} & = 0 \\ f_2(x_1, x_2) = \frac{1}{25}\left(10\, x_2 + 1\right)^2 - x_1 - \frac{3}{2} & = 0. \end{cases} \tag{3.32}$$

On représente en Figure 3.21 les itérées succéssives pour 4 suites avec une initialisation différentes. On remarque qu'il est très difficile, si l'on n'est pas suffisament proche d'un point fixe, de prédire vers lequel on converge.

En Figure 3.22a, on représente les bassins d'attraction pour les itérées de Newton associés au système 3.32 : à chaque point initial $x_0 = (x, y)$ on associe le point fixe vers lequel la suite de Newton converge et chaque point fixe correspond une couleur. En Figure 3.22b, on représente le nombre d'itérations assurant la convergence des itérées de Newton : à chaque point initial $x_0 = (x, y)$ on associe le nombre d'itérations nécessaire à la convergence et une échelle de couleur permet de visualer ces nombres.

Exemple complexe : $z^3 - 1 = 0$

On souhaite trouver les racines complexes de $z^3 - 1$. Pour celà on peut poser $z = x + \imath y$, et le système équivalent devient

SNL:eq:system02}
$$\begin{cases} f_1(x, y) = x^3 - 3xy^2 - 1 & = 0 \\ f_2(x, y) = 3x^2 y - x^3 & = 0. \end{cases} \tag{3.33}$$

Bien évidemment en restant dans le corps des complexes, l'algorithme de Newton est le même (encore faut-il que le langage de programmation utilisé le supporte).

On représente en Figure 3.23 les bassins d'attraction et le nombre d'itérations de convergence associé à des données initiales dans $[-1.5, 1.5] \times [-1.5, 1.5]$. On obtient alors une *fractale de Newton*. Pour illustrer ce *caractère fractale* des représentations, on donne en Figures 3.24 et 3.25 des zooms successifs sur les graphes.

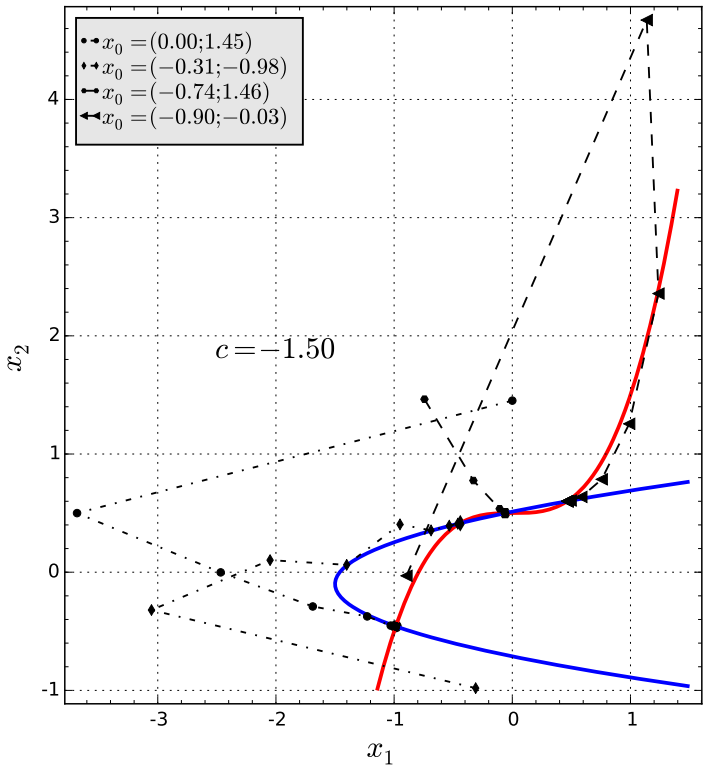
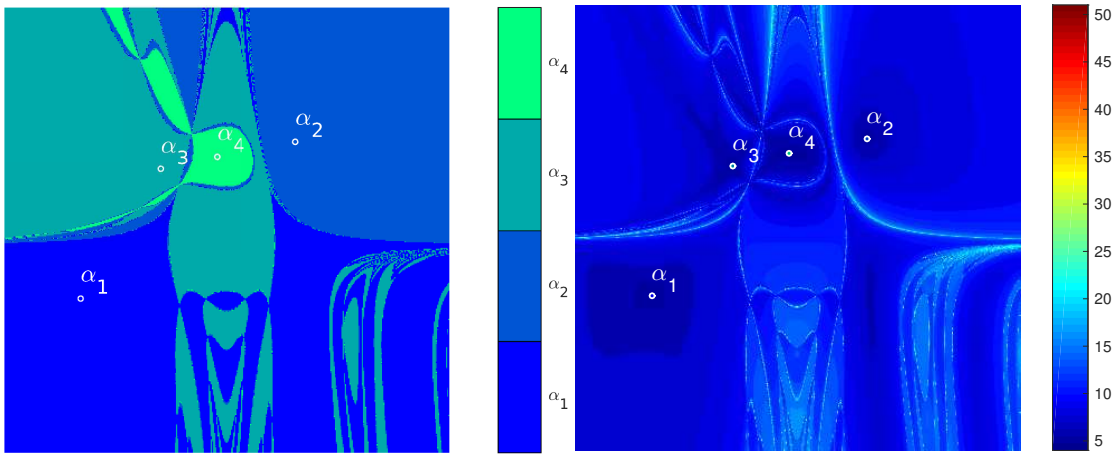


Figure 3.21: Représentation de 4 suites pour le système 3.32 RSNL:fig:system:02

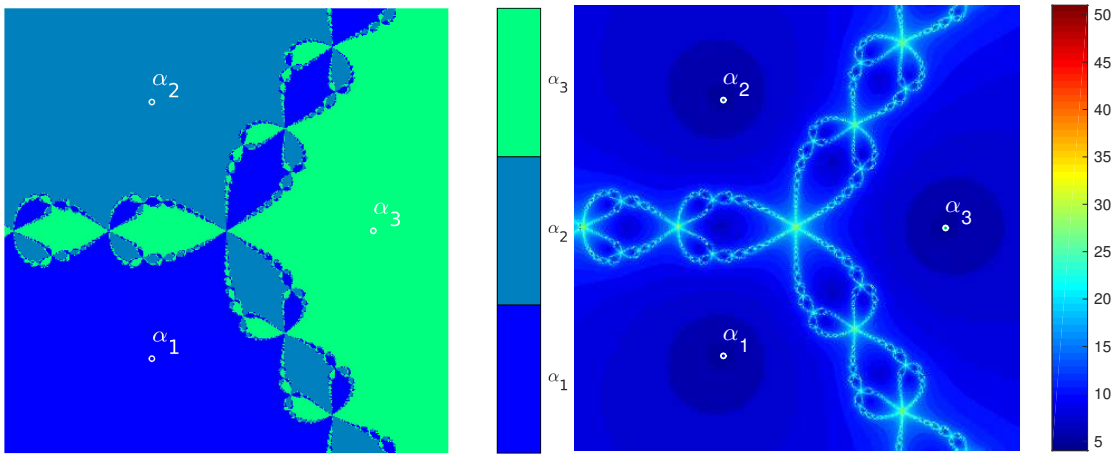


(a) Bassin d'attraction des racines

(b) Nombre d'itérations de convergence

nAttraction01} {subRSNL:Fig:IterNum01}

Figure 3.22: Méthode de Newton, système (3.32) RSNL:Fig:System:01



(a) Bassin d'attraction des racines (b) Nombre d'itérations de convergence

BasssinAttraction02} {subQRSNL:Fig:IterNum02} RSNL:Fig:System:02

Figure 3.23: Méthode de Newton, système (3.33)

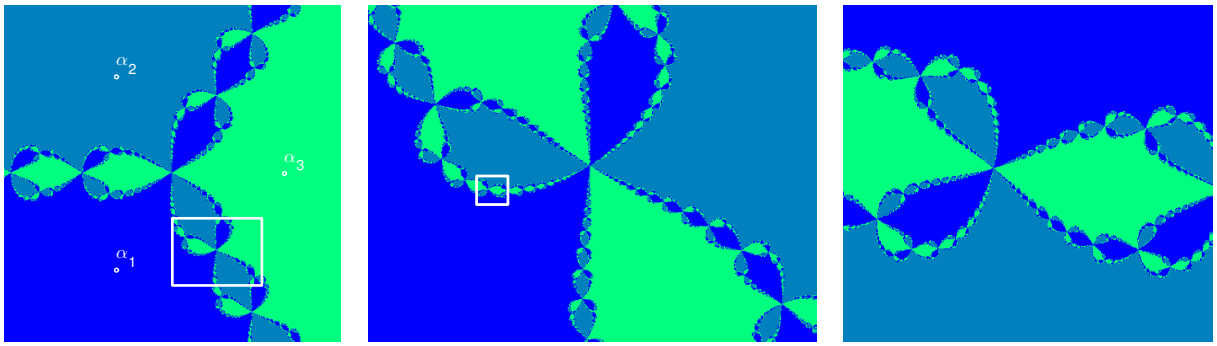


Figure 3.24: Méthode de Newton, système (3.33), zooms sur les bassins d'attraction RSNL:Fig:System02:Bas

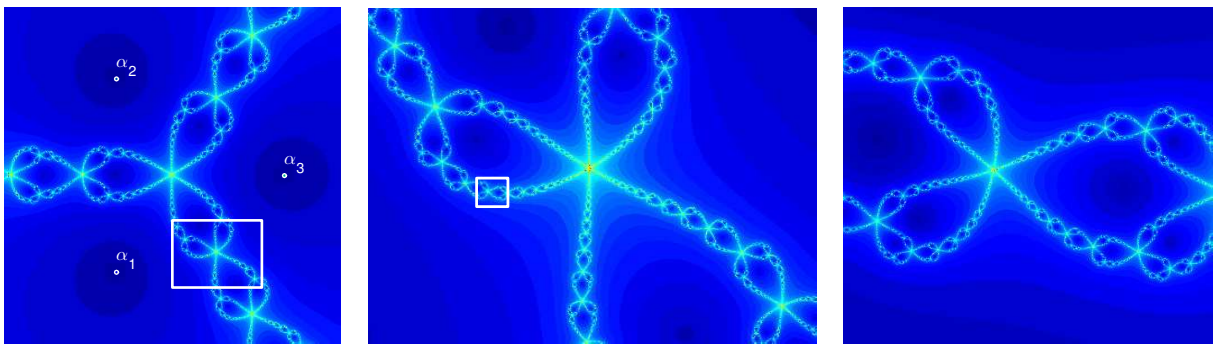
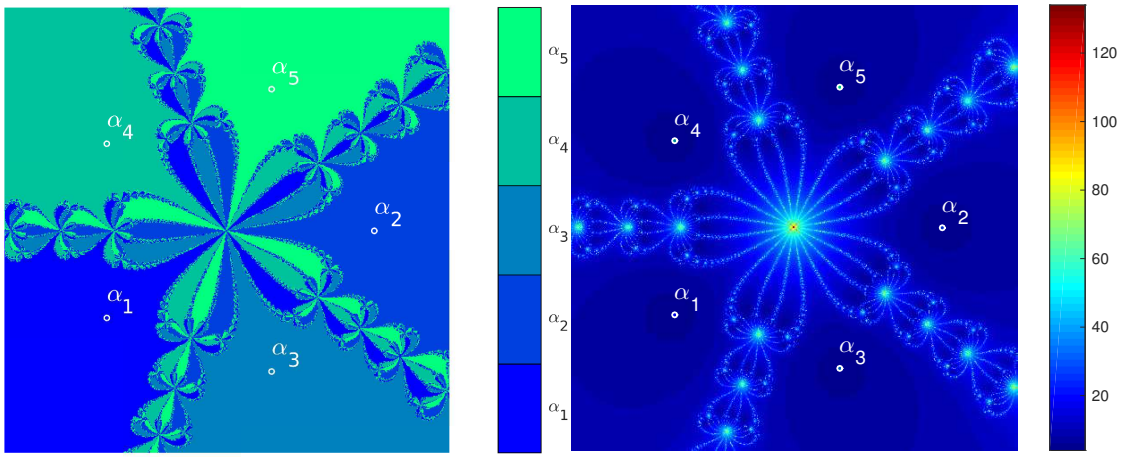


Figure 3.25: Méthode de Newton, système (3.33), zooms sur les nombres d'itérations RSNL:Fig:System02:Ite

Exemple complexe : $z^5 - 1 = 0$

On représente en Figure 3.27 les bassins d'attraction et le nombre d'itérations de convergence associé à des données initiales dans $[-1.5, 1.5] \times [-1.5, 1.5]$.



(a) Bassin d'attraction des racines

(b) Nombre d'itérations de convergence

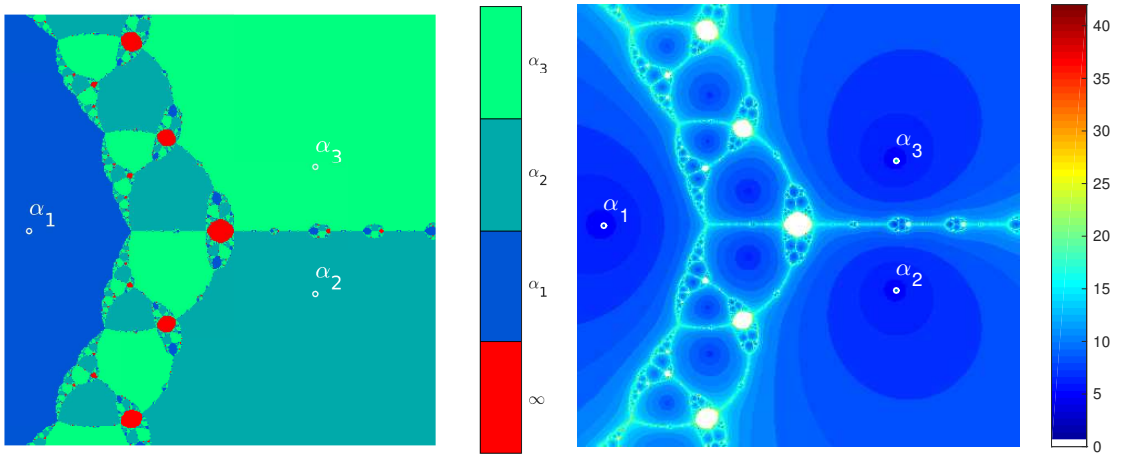
{RSNL:Fig:BasinAttraction03}{RSNL:Fig:NumIter03}

RSNL:Fig:System:03

Figure 3.26: Méthode de Newton pour $z^5 - 1 = 0$

Exemple complexe : $z^3 - 2z + 2 = 0$

On représente en Figure **??** les bassins d'attraction et le nombre d'itérations de convergence associé à des données initiales dans $[-2, 2] \times [-2, 2]$.



(a) Bassin d'attraction des racines. En rouge zône de divergence

(b) Nombre d'itérations de convergence. En blanc zône de divergence

{RSNL:Fig:BasinAttraction04}{RSNL:Fig:NumIter04}

RSNL:Fig:System:03

Figure 3.27: Méthode de Newton pour $z^3 - 2z + 2 = 0$

Chapitre 4

Résolution de systèmes linéaires

Dans cette partie nous allons consisérer la résolution numérique d'un système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$ dont la matrice $\mathbb{A}$ est inversible.

On pourrait penser que pour résoudre le système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$, $\mathbb{A}$ inversible, le plus simple serait de calculer la matrice $\mathbb{A}^{-1}$, inverse de $\mathbb{A}$, puis d'effectuer un produit matrice-vecteur pour obtenir $\boldsymbol{x} = \mathbb{A}^{-1}\boldsymbol{b}$. Or pour calculer l'inverse d'une matrice d'ordre n on doit résoudre n systèmes linéaires d'ordre n! En effet, déterminer l'inverse d'une matrice $\mathbb{A}$ revient à rechercher la matrice $\mathbb{X}$ solution de

$$\mathbb{A}\mathbb{X} = \mathbb{I} \iff \begin{pmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,n} \\ A_{2,1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & A_{n-1,n} \\ A_{n,1} & \dots & A_{n,n-1} & A_{n,n} \end{pmatrix} \begin{pmatrix} X_{1,1} & X_{1,2} & \dots & X_{1,n} \\ X_{2,1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & X_{n-1,n} \\ X_{n,1} & \dots & X_{n,n-1} & X_{n,n} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & 1 \end{pmatrix}$$

Si on note $\boldsymbol{X}_{i,:}$ le i-ème vecteur colonne de la matrice $\mathbb{X}$ et $\boldsymbol{e}_i$ le i-ème de la base canonique de $\mathbb{R}^n$ alors le système précédant s'écrit

$$\begin{pmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,n} \\ A_{2,1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & A_{n-1,n} \\ A_{n,1} & \dots & A_{n,n-1} & A_{n,n} \end{pmatrix} \left(\begin{array}{c|c|c|c} & & & \\ \boldsymbol{X}_1 & \dots & \boldsymbol{X}_n & \\ & & & \end{array} \right) = \left(\begin{array}{c|c|c|c} & & & \\ \boldsymbol{e}_1 & \dots & \boldsymbol{e}_n & \\ & & & \end{array} \right)$$

Ce dernier système est alors équivalent à résoudre les n systèmes linéaires

$$\mathbb{A}\boldsymbol{X}_j = \boldsymbol{e}_j, \quad \forall j \in [\![1, n]\!].$$

> ⚠ Pour résoudre un système linéaire, on ne calcule pas la matrice inverse associée.

Nous allons en section 4.2 étudier quelques **méthodes directes** pour la résolution d'un système linéaire basées sur la recherche d'une matrice $\mathbb{M}$ inversible telle quel la matrice $\mathbb{M}\mathbb{A}$ soit triangulaire supérieure. Ceci conduit à la résolution du système linéaire équivalent

$$\mathbb{M}\mathbb{A}\boldsymbol{x} = \mathbb{M}\boldsymbol{b}.$$

par la *méthode de la remontée* décrite en section 4.2.1).

En section 4.3, nous nous intéresserons aux **méthodes itératives** pour la résolution d'un système linéaire qui peuvent s'écrire sous la forme

$$\boldsymbol{x}^{[k+1]} = \mathbb{B}\boldsymbol{x}^{[k]} + \boldsymbol{c}, \quad k \geqslant 0, \ \boldsymbol{x}^{[0]} \text{ donné}$$

où la matrice $\mathbb{B}$ et le vecteur $\boldsymbol{c}$ sont construits à partir de la matrice $\mathbb{A}$ et du vecteur $\boldsymbol{b}$. On espère alors avoir $\lim_{k\to+\infty} \boldsymbol{x}^{[k]} = \boldsymbol{x}$.

4.1 Exercices et résultats préliminaires

Exercice 4.1.1

{RSL:RP:exo:01}

Soit $\mathbb{A} \in \mathcal{M}_{n,n}(\mathbb{C})$ une matrice et $(\lambda, \boldsymbol{u})$ un élément propre de $\mathbb{A}$ avec $\|\boldsymbol{u}\|_2 = 1$.

Q. 1 *En s'aidant de la base canonique* $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n\}$, *construire une base orthonormée* $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ *telle que* $\boldsymbol{x}_1 = \boldsymbol{u}$.

Notons $\mathbb{P}$ la matrice de changement de base canonique $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n\}$ dans la base $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$:

$$\mathbb{P} = \left(\begin{array}{c|c|c} & & \\ \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_n \\ & & \end{array} \right)$$

Soit $\mathbb{B}$ la matrice définie par $\mathbb{B} = \mathbb{P}^*\mathbb{A}\mathbb{P}$.

Q. 2 1. *Exprimer les coefficients de la matrice* $\mathbb{B}$ *en fonction de la matrice* $\mathbb{A}$ *et des vecteurs* $\boldsymbol{x}_i$, $i \in [\![1, n]\!]$.

$$\mathbb{B} = \mathbb{P}^*\mathbb{A}\mathbb{P}.$$

2. *En déduire que la première colonne de* $\mathbb{B}$ *est* $(\lambda, 0, \ldots, 0)^t$.

Q. 3 *Montrer par récurrence sur l'ordre de la matrice que la matrice* $\mathbb{A}$ *s'écrit*

$$\mathbb{A} = \mathbb{U}\mathbb{T}\mathbb{U}^*$$

où $\mathbb{U}$ *est une matrice unitaire et* $\mathbb{T}$ *une matrice triangulaire supérieure.*

Q. 4 *En supposant* $\mathbb{A}$ *inversible et la décomposition* $\mathbb{A} = \mathbb{U}\mathbb{T}\mathbb{U}^*$ *connue, expliquer comment résoudre "simplement" le système linéaire* $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$.

Correction Exercice 4.1.1

Q. 1 La première chose à faire est de construire une base contenant $\boldsymbol{u}$ à partir de la base canonique $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n\}$. Comme le vecteur propre $\boldsymbol{u}$ est non nul, il existe $j \in [\![1, n]\!]$ tel que $\langle \boldsymbol{u}, \boldsymbol{e}_j \rangle \neq 0$. La famille $\{\boldsymbol{u}, \boldsymbol{e}_1, \ldots, \boldsymbol{e}_{j-1}, \boldsymbol{e}_{j+1}, \ldots, \boldsymbol{e}_n\}$ forme alors une base de $\mathbb{C}^n$ car $\boldsymbol{u}$ n'est pas combinaison linéaire des $\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{j-1}, \boldsymbol{e}_{j+1}, \ldots, \boldsymbol{e}_n\}$.
On note $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\}$ la base dont le premier élément est $\boldsymbol{z}_1 = \boldsymbol{u}$:

$$\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\} = \{\boldsymbol{u}, \boldsymbol{e}_1, \ldots, \boldsymbol{e}_{j-1}, \boldsymbol{e}_{j+1}, \ldots, \boldsymbol{e}_n\}.$$

On peut ensuite utiliser le **procédé de Gram-Schmidt**, rappelé en Proposition A.12, pour construire une base orthonormée à partir de cette base.
On calcule successivement les vecteurs $\boldsymbol{x}_i$ à partir de la base $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\}$ en construisant un vecteur $\boldsymbol{w}_i$ orthogonal aux vecteurs $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}$.

$$\boldsymbol{w}_i = \boldsymbol{z}_i - \sum_{k=1}^{i-1} \langle \boldsymbol{z}_i, \boldsymbol{x}_k \rangle \boldsymbol{x}_k$$

puis on obtient le vecteur $\boldsymbol{x}_i$ en normalisant

$$\boldsymbol{x}_i = \frac{\boldsymbol{w}_i}{\|\boldsymbol{w}_i\|}$$

Q. 2 1. En conservant l'écriture colonne de la matrice $\mathbb{P}$ on obtient

$$\mathbb{B} = \left(\begin{array}{c} \boldsymbol{x}_1^* \\ \hline \boldsymbol{x}_2^* \\ \hline \vdots \\ \hline \boldsymbol{x}_n^* \end{array}\right) \mathbb{A} \left(\begin{array}{c|c|c|c} \boldsymbol{x_1} & \boldsymbol{x}_2 & \dots & \boldsymbol{x}_n \end{array}\right) = \left(\begin{array}{c} \boldsymbol{x}_1^* \\ \hline \boldsymbol{x}_2^* \\ \hline \vdots \\ \hline \boldsymbol{x}_n^* \end{array}\right) \left(\begin{array}{c|c|c|c} \mathbb{A}\boldsymbol{x_1} & \mathbb{A}\boldsymbol{x}_2 & \dots & \mathbb{A}\boldsymbol{x}_n \end{array}\right)$$

Ce qui donne

$$\mathbb{B} = \begin{pmatrix} \boldsymbol{x}_1^*\mathbb{A}\boldsymbol{x_1} & \boldsymbol{x_1}^*\mathbb{A}\boldsymbol{x}_2 & \dots & \boldsymbol{x_1}^*\mathbb{A}\boldsymbol{x}_n \\ \boldsymbol{x}_2^*\mathbb{A}\boldsymbol{x_1} & \boldsymbol{x}_2^*\mathbb{A}\boldsymbol{x}_2 & \dots & \boldsymbol{x}_2^*\mathbb{A}\boldsymbol{x}_n \\ \vdots & \vdots & & \vdots \\ \boldsymbol{x}_n^*\mathbb{A}\boldsymbol{x_1} & \boldsymbol{x}_n^*\mathbb{A}\boldsymbol{x}_2 & \dots & \boldsymbol{x}_n^*\mathbb{A}\boldsymbol{x}_n \end{pmatrix}$$

On a donc

$$B_{i,j} = \boldsymbol{x}_i^*\mathbb{A}\boldsymbol{x_j}, \quad \forall (i,j) \in [\![1,n]\!]^2$$

.

2. On a $\mathbb{A}\boldsymbol{u} = \lambda\boldsymbol{u}$, $\|\boldsymbol{u}\| = 1$, la base $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}$ est orthonormée et $\boldsymbol{x}_1 = \boldsymbol{u}$. on obtient alors

$$\mathbb{B} = \left(\begin{array}{c|c|c|c} \lambda\boldsymbol{u}^*\boldsymbol{u} & \boldsymbol{u}^*\mathbb{A}\boldsymbol{x}_2 & \dots & \boldsymbol{u}^*\mathbb{A}\boldsymbol{x}_n \\ \hline \lambda\boldsymbol{x}_2^*\boldsymbol{u} & \boldsymbol{x}_2^*\mathbb{A}\boldsymbol{x}_2 & \dots & \boldsymbol{x}_2^*\mathbb{A}\boldsymbol{x}_n \\ \hline \vdots & \vdots & & \vdots \\ \hline \lambda\boldsymbol{x}_n^*\boldsymbol{u} & \boldsymbol{x}_n^*\mathbb{A}\boldsymbol{x}_2 & \dots & \boldsymbol{x}_n^*\mathbb{A}\boldsymbol{x}_n \end{array}\right) = \left(\begin{array}{c|c|c|c} \lambda & \boldsymbol{u}^*\mathbb{A}\boldsymbol{x}_2 & \dots & \boldsymbol{u}^*\mathbb{A}\boldsymbol{x}_n \\ \hline 0 & \boldsymbol{x}_2^*\mathbb{A}\boldsymbol{x}_2 & \dots & \boldsymbol{x}_2^*\mathbb{A}\boldsymbol{x}_n \\ \hline \vdots & \vdots & & \vdots \\ \hline 0 & \boldsymbol{x}_n^*\mathbb{A}\boldsymbol{x}_2 & \dots & \boldsymbol{x}_n^*\mathbb{A}\boldsymbol{x}_n \end{array}\right)$$

Q. 3 On veut démontrer, par récurrence faible, la proposition suivante pour $n \geqslant 2$

$(\mathcal{P}_n)$ $\forall \mathbb{A} \in \mathcal{M}_n(\mathbb{C})$, $\exists \mathbb{U} \in \mathcal{M}_n(\mathbb{C})$ unitaire, $\exists \mathbb{T} \in \mathcal{M}_n(\mathbb{C})$ triangulaire supérieure, telles que $\mathbb{A} = \mathbb{U}\mathbb{T}\mathbb{U}^*$.

Initialisation : Montrons que $(\mathcal{P}_2)$ est vérifié.

Soit $\mathbb{A}_2 \in \mathcal{M}_2(\mathbb{C})$. Elle admet au moins un élément propre $(\lambda, \boldsymbol{u})$ (voir Proposition A.39 par ex.) avec $\|\boldsymbol{u}\| = 1$. On peut donc appliquer le résultat de la question précédente : il existe une matrice unitaire $\mathbb{P}_2 \in \mathcal{M}_2(\mathbb{C})$ telle que la matrice $\mathbb{B}_2 = \mathbb{P}_2\mathbb{A}_2\mathbb{P}_2^*$ ait comme premier vecteur colonne $(\lambda, 0)^t$. La matrice $\mathbb{B}_2$ est donc triangulaire supérieure et comme $\mathbb{P}_2$ est unitaire on en déduit

$$\mathbb{A}_2 = \mathbb{P}_2^*\mathbb{B}_2\mathbb{P}_2.$$

On pose $\mathbb{U}_2 = \mathbb{P}_2^*$ matrice unitaire et $\mathbb{T}_2 = \mathbb{B}_2$ matrice triangulaire supérieure pour conclure que la propostion $(\mathcal{P}_2)$ est vraie.

Hérédité : Supposons que $(\mathcal{P}_{n-1})$ soit vérifiée. Montrons que $(\mathcal{P}_n)$ est vraie.

Soit $\mathbb{A}_n \in \mathcal{M}_n(\mathbb{C})$. Elle admet au moins un élément propre $(\lambda, \boldsymbol{u})$ (voir Proposition A.39 par ex.) avec $\|\boldsymbol{u}\| = 1$. On peut donc appliquer le résultat de la question précédente : il existe une matrice unitaire $\mathbb{P}_n \in \mathcal{M}_n(\mathbb{C})$ telle que la matrice $\mathbb{B}_n = \mathbb{P}_n\mathbb{A}_n\mathbb{P}_n^*$ s'écrivent

$$\mathbb{B}_n = \left(\begin{array}{c|c} \lambda & \boldsymbol{c}_{n-1}^* \\ \hline 0 & \\ \vdots & \mathbb{A}_{n-1} \\ 0 & \end{array}\right)$$

où $\boldsymbol{c}_{n-1} \in \mathcal{M}_{n-1}(\mathbb{C})$ et $\mathbb{A}_{n-1} \in \mathcal{M}_{n-1}(\mathbb{C})$. Par hypothèse de récurrence, $\exists \mathbb{U}_{n-1} \in \mathcal{M}_{n-1}(\mathbb{C})$ unitaire et $\mathbb{T}_{n-1} \in \mathcal{M}_{n-1}(\mathbb{C})$ triangulaire supérieure telles que

$$\mathbb{A}_{n-1} = \mathbb{U}_{n-1}\mathbb{T}_{n-1}\mathbb{U}_{n-1}^*$$

ou encore

$$\mathbb{T}_{n-1} = \mathbb{U}_{n-1}^*\mathbb{A}_{n-1}\mathbb{U}_{n-1}.$$

Soit $\mathbb{Q}_n \in \mathcal{M}_n(\mathbb{C})$ la matrice définie par

$$\mathbb{Q}_n = \left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline 0 & & & \\ \vdots & & \mathbb{U}_{n-1} & \\ 0 & & & \end{array}\right).$$

La matrice $\mathbb{Q}_n$ est unitaire. En effet on a

$$\mathbb{Q}_n\mathbb{Q}_n^* = \left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline 0 & & & \\ \vdots & & \mathbb{U}_{n-1} & \\ 0 & & & \end{array}\right)\left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline 0 & & & \\ \vdots & & \mathbb{U}_{n-1}^* & \\ 0 & & & \end{array}\right) = \left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline 0 & & & \\ \vdots & & \underbrace{\mathbb{U}_{n-1}\mathbb{U}_{n-1}^*}_{=\mathbb{I}_{n-1}} & \\ 0 & & & \end{array}\right) = \mathbb{I}_n.$$

On note $\mathbb{T}_n$ la matrice définie par $\mathbb{T}_n = \mathbb{Q}_n^*\mathbb{B}_n\mathbb{Q}_n$. On a alors

$$\mathbb{T}_n = \left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline 0 & & & \\ \vdots & & \mathbb{U}_{n-1}^* & \\ 0 & & & \end{array}\right)\left(\begin{array}{c|ccc} \lambda & & \boldsymbol{c}_{n-1}^* & \\ \hline 0 & & & \\ \vdots & & \mathbb{A}_{n-1} & \\ 0 & & & \end{array}\right)\left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline 0 & & & \\ \vdots & & \mathbb{U}_{n-1} & \\ 0 & & & \end{array}\right)$$

$$= \left(\begin{array}{c|ccc} \lambda & & \boldsymbol{c}_{n-1}^* & \\ \hline 0 & & & \\ \vdots & & \mathbb{U}_{n-1}^*\mathbb{A}_{n-1} & \\ 0 & & & \end{array}\right)\left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline 0 & & & \\ \vdots & & \mathbb{U}_{n-1} & \\ 0 & & & \end{array}\right) = \left(\begin{array}{c|ccc} \lambda & & \boldsymbol{c}_{n-1}^*\mathbb{U}_{n-1}^* & \\ \hline 0 & & & \\ \vdots & & \underbrace{\mathbb{U}_{n-1}^*\mathbb{A}_{n-1}\mathbb{U}_{n-1}}_{=\mathbb{T}_{n-1}} = & \\ 0 & & & \end{array}\right)$$

La matrice $\mathbb{T}_n$ est donc triangulaire supérieure et on a par définition de $\mathbb{B}_n$

$$\mathbb{T}_n = \mathbb{Q}_n^*\mathbb{P}_n\mathbb{A}_n\mathbb{P}_n^*\mathbb{Q}_n.$$

On note $\mathbb{U}_n = \mathbb{P}_n^*\mathbb{Q}_n$. Cette matrice est unitaire car les matrices $\mathbb{Q}_n$ et $\mathbb{P}_n$ le sont. En effet, on a

$$\mathbb{U}_n\mathbb{U}_n^* = \mathbb{P}_n^*\mathbb{Q}_n(\mathbb{P}_n^*\mathbb{Q}_n)^* = \mathbb{P}_n^*\underbrace{\mathbb{Q}_n\mathbb{Q}_n^*}_{=\mathbb{I}_n}\mathbb{P}_n = \mathbb{P}_n^*\mathbb{P}_n = \mathbb{I}_n.$$

On a $\mathbb{T}_n = \mathbb{U}_n^*\mathbb{A}_n\mathbb{U}_n$ et en multipliant cette équation à gauche par $\mathbb{U}_n$ et à droite par $\mathbb{U}_n^*$ on obtient l'équation équivalente $\mathbb{A}_n = \mathbb{U}_n\mathbb{T}_n\mathbb{U}_n^*$. La propriété $(\mathcal{P}_n)$ est donc vérifiée. Ce qui achève la démonstration.

Q. 4 Résoudre $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$ est équivalent à résoudre

SI.exo04v2:eq10}
$$\mathbb{U}\mathbb{T}\mathbb{U}^*\boldsymbol{x} = \boldsymbol{b}. \tag{4.1}$$

Comme $\mathbb{U}$ est unitaire, on a $\mathbb{U}\mathbb{U}^* = \mathbb{I}$ et $\mathbb{U}^*$ inversible. Donc en multipliant (4.1) par $\mathbb{U}^*$ on obtient le système équivalent

$$\underbrace{\mathbb{U}^*\mathbb{U}}_{=\mathbb{I}}\mathbb{T}\mathbb{U}^*\boldsymbol{x} = \mathbb{U}^*\boldsymbol{b} \iff \mathbb{T}\mathbb{U}^*\boldsymbol{x} = \mathbb{U}^*\boldsymbol{b}.$$

On pose $\boldsymbol{y} = \mathbb{U}^*\boldsymbol{x}$. Le système précédant se résoud en deux étapes

1. on cherche $\boldsymbol{y}$ solution de $\mathbb{T}\boldsymbol{y} = \mathbb{U}^*\boldsymbol{b}$. Comme $\mathbb{U}$ est unitaire on a $\det(\mathbb{U})\det(\mathbb{U}^*) = \det(\mathbb{I}) = 1$ et donc
$$\begin{aligned}\det(\mathbb{A}) &= \det(\mathbb{U}\mathbb{T}\mathbb{U}^*) = \det(\mathbb{U})\det(\mathbb{T})\det(\mathbb{U}^*)\\ &= \det(\mathbb{T})\end{aligned}$$
Or $\mathbb{A}$ inversible équivalent à $\det(\mathbb{A}) \neq 0$ et donc la matrice $\mathbb{T}$ est inversible. La matrice $\mathbb{T}$ étant triangulaire inférieure on peut résoudre facilement le système par la *méthode de remontée*.
2. une fois $\boldsymbol{y}$ déterminé, on résoud $\mathbb{U}^*\boldsymbol{x} = \boldsymbol{y}$. Comme $\mathbb{U}$ est unitaire, on obtient directement $\boldsymbol{x} = \mathbb{U}\boldsymbol{y}$.

◇

On tire de cet exercice le théorème suivant

Théorème 4.1:

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$. Il existe une matrice unitaire $\mathbb{U}$ et une matrice triangulaire supérieure $\mathbb{T}$ telles que

$$\mathbb{A} = \mathbb{U}\mathbb{T}\mathbb{U}^* \tag{4.2}$$

Proof. voir Exercice 4.1.1 □

Exercice 4.1.2: Matrice d'élimination

Soit $\boldsymbol{v} \in \mathbb{C}^n$ avec $v_1 \neq 0$. On note $\mathbb{E}^{[\boldsymbol{v}]} \in \mathcal{M}_n(\mathbb{C})$ la matrice triangulaire inférieure à diagonale unité définie par

$$\mathbb{E}^{[\boldsymbol{v}]} = \left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline -v_2/v_1 & 1 & 0 & \dots \\ \vdots & 0 & \ddots & \ddots \\ \vdots & \vdots & \ddots & 0 \\ -v_n/v_1 & 0 & \dots & 1 \end{array}\right) \tag{4.3}$$

Q. 1 *1. Calculer le déterminant de* $\mathbb{E}^{[\boldsymbol{v}]}$.

2. Déterminer l'inverse de $\mathbb{E}^{[\boldsymbol{v}]}$.

$\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ avec $A_{1,1} \neq 0$. On note $\boldsymbol{A}_{:,j}$ le $j$ème vecteur colonne de $\mathbb{A}$ et $\boldsymbol{A}_{i,:}$ son $i$ème vecteur ligne. On pose $\boldsymbol{A}_1 = \boldsymbol{A}_{:,1}$.

Q. 2 *1. Calculer* $\tilde{\mathbb{A}} = \mathbb{E}^{[\boldsymbol{A}_1]}\mathbb{A}$ *en fonction des vecteurs lignes de* $\mathbb{A}$.

2. Montrer que la première colonne de $\tilde{\mathbb{A}}$ *est le vecteur* $(A_{1,1}, 0, \dots, 0)^t$ *i.e.*

$$\mathbb{E}^{[\boldsymbol{A}_1]}\mathbb{A}\boldsymbol{e}_1 = A_{1,1}\boldsymbol{e}_1 \tag{4.4}$$

où $\boldsymbol{e}_1$ *est le premier vecteur de la base canonique de* $\mathbb{C}^n$.

Soit $m \in \mathbb{N}^*$. On note $\mathbb{E}^{[m,\boldsymbol{v}]} \in \mathcal{M}_{m+n}(\mathbb{C})$ la matrice triangulaire inférieure à diagonale unité définie par

$$\mathbb{E}^{[m,\boldsymbol{v}]} = \left(\begin{array}{c|c} \mathbb{I}_m & \mathbb{O} \\ \hline \mathbb{O} & \mathbb{E}^{[\boldsymbol{v}]} \end{array}\right) \tag{4.5}$$

Q. 3 *1. Calculer le déterminant de* $\mathbb{E}^{[m,\boldsymbol{v}]}$.

2. Déterminer l'inverse de $\mathbb{E}^{[m,\boldsymbol{v}]}$ *en fonction de l'inverse de* $\mathbb{E}^{[\boldsymbol{v}]}$.

Soit $\mathbb{C}$ la matrice bloc définie par

$$\mathbb{C} = \left(\begin{array}{c|c} \mathbb{C}_{1,1} & \mathbb{C}_{1,2} \\ \hline \mathbb{O} & \mathbb{A} \end{array}\right)$$

où $\mathbb{C}_{1,1} \in \mathcal{M}_m(\mathbb{C})$ et $\mathbb{C}_{1,2} \in \mathcal{M}_{m,n}(\mathbb{C})$.

Q. 4 *Déterminer la matrice produit* $\mathbb{E}^{[m,\boldsymbol{A}_1]}\mathbb{B}$ *en fonction des matrices* $\mathbb{C}_{1,1}$, $\mathbb{C}_{1,2}$ *et* $\tilde{\mathbb{A}}$.

Correction Exercice 4.1.2

Q. 1 1. La matrice $\mathbb{E}^{[\boldsymbol{v}]}$ est triangulaire : son déterminant est donc le produit de ses éléments diagonaux (voir Proposition A.46, page 122). On a alors $\det(\mathbb{E}^{[\boldsymbol{v}]}) = 1$.

2. Pour calculer son inverse qui existe puisque $\det(\mathbb{E}^{[\boldsymbol{v}]}) \neq 0$, on écrit $\mathbb{E}^{[\boldsymbol{v}]}$ sous forme bloc :

$$\mathbb{E}^{[\boldsymbol{v}]} = \left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline \boldsymbol{e} & & \mathbb{I}_{n-1} & \end{array}\right)$$

avec $\boldsymbol{e} = (-v_2/v_1, \dots, -v_n/v_1)^{\mathrm{t}} \in \mathbb{C}^{n-1}$ On note $\mathbb{X} \in \mathcal{M}_n(\mathbb{C})$ son inverse qui s'écrit avec la même structure bloc

$$\mathbb{X} = \left(\begin{array}{c|c} a & \boldsymbol{b}^* \\ \hline \boldsymbol{c} & \mathbb{D} \end{array}\right)$$

avec $a \in \mathbb{K}$, $\boldsymbol{b} \in \mathbb{K}^{n-1}$, $\boldsymbol{c} \in \mathbb{K}^{n-1}$ et $\mathbb{D} \in \mathcal{M}_{n-1}(\mathbb{C})$.
La matrice $\mathbb{X}$ est donc solution de $\mathbb{E}^{[\boldsymbol{v}]}\mathbb{X} = \mathbb{I}$. Grace à l'écriture bloc des matrices on en déduit rapidement la matrice $\mathbb{X}$. En effet, en utilisant les produits blocs des matrices, on obtient

$$\mathbb{E}^{[\boldsymbol{v}]}\mathbb{X} = \left(\begin{array}{c|c} 1 & \mathbf{0}_{n-1}^{\mathrm{t}} \\ \hline \boldsymbol{e} & \mathbb{I}_{n-1} \end{array}\right)\left(\begin{array}{c|c} a & \boldsymbol{b}^* \\ \hline \boldsymbol{c} & \mathbb{D} \end{array}\right) = \left(\begin{array}{c|c} 1 \times a & 1 \times \boldsymbol{b}^* + \mathbf{0}_{n-1}^{\mathrm{t}} \times \mathbb{D} \\ \hline \boldsymbol{e} \times a + \mathbb{I}_{n-1} \times \boldsymbol{c} & \boldsymbol{e} \times \boldsymbol{b}^* + \mathbb{I}_{n-1} \times \mathbb{D} \end{array}\right)$$

$$= \left(\begin{array}{c|c} a & \boldsymbol{b}^* \\ \hline a\boldsymbol{e} + \boldsymbol{c} & \boldsymbol{e}\boldsymbol{b}^* + \mathbb{D} \end{array}\right)$$

Comme $\mathbb{X}$ est l'inverse de $\mathbb{E}^{[\boldsymbol{v}]}$, on a $\mathbb{E}^{[\boldsymbol{v}]}\mathbb{X} = \mathbb{I}$ et donc en écriture bloc

$$\left(\begin{array}{c|c} a & \boldsymbol{b}^* \\ \hline a\boldsymbol{e} + \boldsymbol{c} & \boldsymbol{e}\boldsymbol{b}^* + \mathbb{D} \end{array}\right) = \left(\begin{array}{c|c} 1 & \mathbf{0}_{n-1}^{\mathrm{t}} \\ \hline \mathbf{0}_{n-1} & \mathbb{I}_{n-1} \end{array}\right).$$

Ceci revient à résoudre les 4 équations

$$a = 1, \quad \boldsymbol{b}^* = \mathbf{0}_{n-1}^{\mathrm{t}}, \quad a\boldsymbol{e} + \boldsymbol{c} = \mathbf{0}_{n-1} \quad \text{et} \quad \boldsymbol{e}\boldsymbol{b}^* + \mathbb{D} = \mathbb{I}_{n-1}$$

qui donnent immédiatement $a = 1$, $\boldsymbol{b} = \mathbf{0}_{n-1}$, $\boldsymbol{c} = -\boldsymbol{e}$ et $\mathbb{D} = \mathbb{I}_{n-1}$. On obtient le résultat suivant

$$\left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline -\boldsymbol{e} & & \mathbb{I}_{n-1} & \end{array}\right)\left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline \boldsymbol{e} & & \mathbb{I}_{n-1} & \end{array}\right) = \mathbb{I}_n.$$


Il aurait été plus rapide d'utiliser la Proposition A.47, page 122.

Q. 2 1. Pour simplifier les notations, on note $\mathbb{E} = \mathbb{E}^{[\boldsymbol{A}_1]}$. Par définition du produit de deux matrices on a

$$\tilde{A}_{i,j} = \sum_{k=1}^{n} E_{i,k}A_{k,j}, \quad \forall (i,j) \in [\![1,n]\!]^2.$$

Quand $i = 1$, on a par construction $E_{1,k} = \delta_{1,k}$ et donc

$$\tilde{A}_{1,j} = A_{1,j}, \ \forall j \in [\![1,n]\!] \iff \tilde{\boldsymbol{A}}_{1,:} = \boldsymbol{A}_{1,:}. \tag{4.6}$$

Pour $i \geqslant 2$, on a $E_{i,1} = -\frac{v_i}{v_1}$ et $E_{i,k} = \delta_{i,k}$, $\forall k \in [\![2,n]\!]$. On obtient alors pour tout $j \in [\![1,n]\!]$

$$\tilde{A}_{i,j} = E_{i,1}A_{1,j} + \sum_{k=2}^{n} E_{i,k}A_{k,j} = -\frac{v_i}{v_1}A_{1,j} + \sum_{k=2}^{n} \delta_{i,k}A_{k,j} = -\frac{v_i}{v_1}A_{1,j} + A_{i,j}$$

ce qui donne pour tout $i \in [\![2,n]\!]$

$$\tilde{A}_{i,j} = A_{i,j} - \frac{v_i}{v_1}A_{1,j}, \ \forall j \in [\![1,n]\!] \iff \tilde{\boldsymbol{A}}_{i,:} = -\frac{v_i}{v_1}\boldsymbol{A}_{1,:} + \boldsymbol{A}_{i,:} \tag{4.7}$$

En conclusion, la matrice $\tilde{\mathbb{A}}$ s'écrit

$$\tilde{\mathbb{A}} = \begin{pmatrix} \boldsymbol{A}_{1,:} \\ \hdashline \boldsymbol{A}_{2,:} - (v_2/v_1)\boldsymbol{A}_{1,:} \\ \hdashline \vdots \\ \hdashline \boldsymbol{A}_{n,:} - (v_n/v_1)\boldsymbol{A}_{1,:} \end{pmatrix}$$

2. De (4.6), on tire $\tilde{A}_{1,1} = A_{1,1}$. A partir de (4.7) on obtient pour tout $i \in [\![2, n]\!]$, $\tilde{A}_{i,1} = A_{i,1} - \frac{v_i}{v_1} A_{1,1}$. Par construction $v_j = A_{j,1}$ pour tout $j \in [\![1, n]\!]$, ce qui donne $\tilde{A}_{i,1} = 0$. La première colonne de $\tilde{\mathbb{A}}$ est $(1, 0, \ldots, 0)^{\mathbf{t}}$.

Q. 3 1. La matrice $\mathbb{E}^{[m,\boldsymbol{v}]}$ est triangulaire inférieure. Son déterminant est donc le produit de ses éléments diagonaux. Comme cette matrice est à diagonale unité (i.e. tous ses éléments diagonaux valent 1), on obtient $\det \mathbb{E}^{[m,\boldsymbol{v}]} = 1$.

Une autre manière de le démontrer. On peut voir que la matrice $\mathbb{E}^{[m,\boldsymbol{v}]}$ est bloc-diagonale. D'après la Proposition A.47, page 122, son déterminant est le produit des déterminant des blocs diagonaux : $\det \mathbb{E}^{[m,\boldsymbol{v}]} = \det \mathbb{I}_m \times \det \mathbb{E}^{[\boldsymbol{v}]} = 1$.

2. On note $\mathbb{X}$ l'inverse de la matrice $\mathbb{E}^{[m,\boldsymbol{v}]}$. Cette matrice s'écrit avec la même structure bloc

$$\mathbb{X}\left(\begin{array}{c|c} \mathbb{X}_{1,1} & \mathbb{X}_{1,2} \\ \hline \mathbb{X}_{2,1} & \mathbb{X}_{2,2} \end{array}\right) \text{ avec } \mathbb{X}_{1,1} \in \mathcal{M}_m(\mathbb{C}) \text{ et } \mathbb{X}_{2,2} \in \mathcal{M}_n(\mathbb{C})$$

On a donc $\mathbb{X}\mathbb{E}^{[m,\boldsymbol{v}]} = \mathbb{I}_{m+n}$ c'est à dire en écriture bloc

$$\left(\begin{array}{c|c} \mathbb{X}_{1,1} & \mathbb{X}_{1,2} \\ \hline \mathbb{X}_{2,1} & \mathbb{X}_{2,2} \end{array}\right)\left(\begin{array}{c|c} \mathbb{I}_m & \mathbb{O} \\ \hline \mathbb{O} & \mathbb{E}^{[\boldsymbol{v}]} \end{array}\right) = \left(\begin{array}{c|c} \mathbb{I}_m & \mathbb{O} \\ \hline \mathbb{O} & \mathbb{I}_n \end{array}\right) =$$

On doit donc résoudre les 4 équations suivantes :

$$\mathbb{X}_{1,1}\mathbb{I}_m = \mathbb{I}_m, \quad \mathbb{X}_{1,2}\mathbb{I}_n = \mathbb{O}, \quad \mathbb{X}_{2,1}\mathbb{I}_m = \mathbb{O} \quad \text{et} \quad \mathbb{X}_{2,2}\mathbb{E}^{[\boldsymbol{v}]} = \mathbb{I}_n.$$

Comme la matrice $\mathbb{E}^{[\boldsymbol{v}]}$ est inversible, on obtient

$$\mathbb{X} = \left(\begin{array}{c|c} \mathbb{I}_m & \mathbb{O} \\ \hline \mathbb{O} & \left(\mathbb{E}^{[\boldsymbol{v}]}\right)^{-1} \end{array}\right)$$

Plus rapidement, comme la matrice $\mathbb{E}^{[m,\boldsymbol{v}]}$ est bloc-diagonale, on en déduit (voir Proposition A.47, page 122) directement le résultat.

Q. 4 Le produit $\mathbb{B}\mathbb{C}$ peut s'effectuer par bloc car les blocs sont de dimensions compatibles et on a

$$\mathbb{B}\mathbb{C} = \left(\begin{array}{c|c} \mathbb{I}_m & \mathbb{O}_{m,n} \\ \hline \mathbb{O}_{n,m} & \mathbb{E} \end{array}\right)\left(\begin{array}{c|c} \mathbb{C}_{1,1} & \mathbb{C}_{1,2} \\ \hline \mathbb{O}_{n,m} & \mathbb{A} \end{array}\right) = \left(\begin{array}{c|c} \mathbb{I}_m\mathbb{C}_{1,1} + \mathbb{O}_{m,n}\mathbb{O}_{n,m} & \mathbb{I}_m\mathbb{C}_{1,2} + \mathbb{O}_{m,n}\mathbb{A} \\ \hline \mathbb{O}_{n,m}\mathbb{C}_{1,1} + \mathbb{E}\mathbb{O}_{n,m} & \mathbb{O}_{n,m}\mathbb{C}_{1,2} + \mathbb{E}\mathbb{A} \end{array}\right)$$
$$= \left(\begin{array}{c|c} \mathbb{C}_{1,1} & \mathbb{C}_{1,2} \\ \hline \mathbb{O}_{n,m} & \mathbb{E}\mathbb{A} \end{array}\right) = \left(\begin{array}{c|c} \mathbb{C}_{1,1} & \mathbb{C}_{1,2} \\ \hline \mathbb{O}_{n,m} & \tilde{\mathbb{A}} \end{array}\right)$$

◇

On tire de cet exercice le lemme suivant

Lemme 4.2

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ avec $A_{1,1} \neq 0$. Il existe une matrice $\mathbb{E} \in \mathcal{M}_n(\mathbb{C})$ triangulaire inférieure à diagonale unité telle que {RSL:lemma:el

$$\mathbb{E}\mathbb{A}\boldsymbol{e}_1 = A_{1,1}\boldsymbol{e}_1 \tag{4.8}$$ {RSL:lemma:el

où $\boldsymbol{e}_1$ est le premier vecteur de la base canonique de $\mathbb{C}^n$.

Exercice 4.1.3: Matrice de permutation

{RSL:RP:exo:03}

Soit $(i,j) \in [\![1,n]\!]^2$, on note $\mathbb{P}_n^{[i,j]} \in \mathcal{M}_n(\mathbb{C})$ la matrice identitée dont on a permuté les lignes i et j.

Q. 1 *Définir proprement cette matrice et la représenter.*

Soient $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ et $\mathbb{B} \in \mathcal{M}_n(\mathbb{C})$. On note $\boldsymbol{A}_{r,:}$ le $r$ème vecteur ligne de $\mathbb{A}$ et $\boldsymbol{B}_{:,s}$ le $s$ème vecteur colonne de $\mathbb{B}$.

Q. 2 1. *Déterminer $\mathbb{P}_n^{[i,j]}\mathbb{A}$ en fonction des vecteurs lignes de $\mathbb{A}$.*

2. *Déterminer $\mathbb{B}\mathbb{P}_n^{[i,j]}$ en fonction des vecteurs colonnes de $\mathbb{B}$.*

Q. 3 1. *Calculer le déterminant de $\mathbb{P}_n^{[i,j]}$.*

2. *Déterminer l'inverse de $\mathbb{P}_n^{[i,j]}$.*

Correction Exercice 4.1.3 On note $\mathbb{P} = \mathbb{P}_n^{[i,j]}$.

Q. 1 On peut définir cette matrice par ligne, $\forall s \in [\![1,n]\!]$,

$$\begin{cases} P_{r,s} &= \delta_{r,s}, \quad \forall r \in [\![1,n]\!]\backslash\{i,j\}, \\ P_{i,s} &= \delta_{j,s}, \\ P_{j,s} &= \delta_{i,s}. \end{cases}$$

On peut noter que la matrice $\mathbb{P}$ est symétrique.

Q. 2 1. On note $\mathbb{D} = \mathbb{P}\mathbb{A}$. Par définition du produit matriciel on a

$$D_{r,s} = \sum_{k=1}^{n} P_{r,k}A_{k,s}.$$

⚠ Ne pas utiliser les indices i et j qui sont déjà fixés dans la définition de la matrice $\mathbb{P} = \mathbb{P}_n^{[i,j]}$.

On obtient, $\forall s \in [\![1,n]\!]$,

$$\begin{cases} D_{r,s} &= \displaystyle\sum_{k=1}^{n} \delta_{r,k}A_{k,s} = A_{r,s}, \quad \forall r \in [\![1,n]\!]\backslash\{i,j\}, \\ D_{i,s} &= \displaystyle\sum_{k=1}^{n} \delta_{j,k}A_{k,s} = A_{j,s}, \\ D_{j,s} &= \displaystyle\sum_{k=1}^{n} \delta_{i,k}A_{k,s} = A_{i,s}. \end{cases}$$

ce qui donne

$$\begin{cases} \boldsymbol{D}_{r,:} &= \boldsymbol{A}_{r,:}, \quad \forall r \in [\![1,n]\!]\backslash\{i,j\}, \\ \boldsymbol{D}_{i,:} &= \boldsymbol{A}_{j,:}, \\ \boldsymbol{D}_{j,:} &= \boldsymbol{A}_{i,:}. \end{cases}$$

2. On note $\mathbb{E} = \mathbb{A}\mathbb{P}$. Par définition du produit matriciel et par symétrie de $\mathbb{P}$ on a

$$E_{r,s} = \sum_{k=1}^{n} A_{r,k}P_{k,s} = \sum_{k=1}^{n} A_{r,k}P_{s,k}.$$

On obtient en raisonnant par colonne, $\forall r \in [\![1,n]\!]$,

$$\left\{\begin{array}{rcl} E_{r,s} & = & \displaystyle\sum_{k=1}^{n} A_{r,k}\delta_{s,k} = A_{r,s}, \quad \forall s \in [\![1,n]\!]\backslash\{i,j\}, \\ E_{r,i} & = & \displaystyle\sum_{k=1}^{n} A_{r,k}\delta_{j,k} = A_{r,j}, \\ E_{r,j} & = & \displaystyle\sum_{k=1}^{n} A_{r,k}\delta_{i,k} = A_{r,i}. \end{array}\right.$$

ce qui donne

$$\left\{\begin{array}{lll} \boldsymbol{E}_{:,s} & = \boldsymbol{A}_{:,s}, & \forall s \in [\![1,n]\!]\backslash\{i,j\}, \\ \boldsymbol{E}_{:,i} & = \boldsymbol{A}_{:,j}, & \\ \boldsymbol{E}_{:,j} & = \boldsymbol{A}_{:,i}. & \end{array}\right.$$

Q. 3 1. $\det(\mathbb{P}) = -1$, si $i \neq j$ et $\det(\mathbb{P}) = 1$ sinon.

2. Immédiat par calcul direct on a $\mathbb{P}\mathbb{P} = \mathbb{I}$ et donc la matrice $\mathbb{P}$ est inversible et $\mathbb{P}^{-1} = \mathbb{P}$.

◇

4.2 Méthodes directes

{RSL:sec:MD}

Pour résoudre le système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$, nous allons le transformer en un système linéaire triangulaire supérieure équivalent

$$\mathbb{M}\mathbb{A}\boldsymbol{x} = \mathbb{M}\boldsymbol{b}$$

où $\mathbb{M}$ est une matrice inversible telle que $\mathbb{M}\mathbb{A}$ soit triangulaire supérieure. Nous allons voir que ce nouveau système est très facile à résoudre par la *méthode de la remontée*.

Nous étudierons la *méthode de Gauss-Jordan* que nous réécrions sous forme algébrique. Puis nous ferons le lien avec les méthodes utilisant la *factorisation* $\mathbb{L}\mathbb{U}$ et la *factorisation de Cholesky*. Nous finirons par une méthode utilisant la factorisation $\mathbb{Q}\mathbb{R}$ d'une matrice, factorisation qui sera réutilisée pour le calcul de valeurs propres et vecteurs propres en section **??**.

Nous allons tout d'abord regarder quelques cas particuliers : la matrice du système est diagonale, triangulaire inférieure ou triangulaire supérieure.

4.2.1 Matrices particulières

{RSL:subsec:MP}

Matrices diagonales

Soit $\mathbb{A}$ une matrice de $\mathcal{M}_n(\mathbb{R})$ diagonale inversible et $\boldsymbol{b} \in \mathbb{R}^n$. Dans ces cas les coefficients diagonaux de $\mathbb{A}$ sont tous non nuls et l'on a

$$x_i = b_i/A_{i,i}, \quad \forall i \in [\![1,n]\!]. \tag{4.9}$$

On a immédiatement l'algorithme

Algorithme 4.1 Fonction RSLMatDiag permettant de résoudre le système linéaire à matrice diagonale inversible

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}.$$

{algo:RSL:Diag}

Données : $\mathbb{A}$: matrice diagonale de $\mathcal{M}_n(\mathbb{R})$ inversible.
$\boldsymbol{b}$: vecteur de $\mathbb{R}^n$.
Résultat : $\boldsymbol{x}$: vecteur de $\mathbb{R}^n$.

1: **Fonction** $\boldsymbol{x} \leftarrow$ RSLMatDiag($\mathbb{A}, \boldsymbol{b}$)
2: **Pour** $i \leftarrow 1$ **à** n **faire**
3: $x(i) \leftarrow b(i)/A(i,i)$
4: **Fin Pour**
5: **Fin Fonction**

Matrices triangulaires inférieures

Soit $\mathbb{A}$ une matrice de $\mathcal{M}_n(\mathbb{R})$ triangulaire inférieure inversible et $\boldsymbol{b} \in \mathbb{R}^n$. On veut résoudre le système linéaire

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \begin{pmatrix} A_{1,1} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ A_{n,1} & \dots & \dots & A_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ \vdots \\ b_n \end{pmatrix}$$

Exercice 4.2.1

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ une matrice **triangulaire**. Montrer que

$$\mathbb{A} \text{ inversible} \iff A_{i,i} \neq 0, \forall i \in [\![1, n]\!].$$

On remarque que l'on peut calculer successivement $x_1, x_1, \dots, x_n$, car il est possible de calculer x_i si on connait $x_1, \dots, x_{i-1}$: c'est la **méthode de descente**. En effet, on a

$$(\mathbb{A}\boldsymbol{x})_i = b_i, \ \forall i \in [\![1, n]\!].$$

et donc, par définition d'un produit matrice-vecteur,

$$\sum_{j=1}^{n} A_{i,j} x_j = b_i, \ \forall i \in [\![1, n]\!].$$

Comme $\mathbb{A}$ est une matrice triangulaire inférieure, on a (voir Définition A.37) $A_{i,j} = 0$ si $j > i$. Ceci donne alors pour tout $i \in [\![1, n]\!]$

$$\begin{aligned} b_i &= \sum_{j=1}^{i-1} A_{i,j} x_j + A_{i,i} x_i + \sum_{j=i+1}^{n} \underbrace{A_{i,j}}_{=0} x_j \\ &= \sum_{j=1}^{i-1} A_{i,j} x_j + A_{i,i} x_i \end{aligned}$$

De plus la matrice $\mathbb{A}$ étant triangulaire inversible ses éléments diagonaux sont tous non nuls. On obtient alors x_i en fonction des $x_{i+1}, \dots, x_n$:

$$x_i = \frac{1}{A_{i,i}} \left(b_i - \sum_{j=1}^{i-1} A_{i,j} x_j \right), \ \forall i \in [\![1, n]\!]. \tag{4.10}$$

{MD:MP:TI:eq03}

On écrit en détail les raffinements successifs permettant d'aboutir à l'Algorithme 4.2 final ne comportant que des opérations élémentaires(.... à finaliser) de telle sorte que le passage entre deux raffinements successifs soit le plus compréhensible possible.

Algorithme 4.2 $\mathcal{R}_0$

1: Résoudre $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$ en calculant successivement $x_1, x_2, \dots, x_n$.

Algorithme 4.2 $\mathcal{R}_1$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $\quad x_i \leftarrow \frac{1}{A_{i,i}} \left(b_i - \sum_{j=1}^{i-1} A_{i,j} x_j \right)$
3: **Fin Pour**

Dans le raffinement $\mathcal{R}_1$, la seule difficulté restante est le calcul de la somme $\sum_{j=1}^{i-1} A_{i,j} x_j$. En effet, l'opérateur mathématique $\sum$ n'est pas défini dans notre langage algorithmique : il va donc falloir détailler un peu plus l'algorithme. Pour isoler le calcul de cette somme, on la note S. La ligne 2 peut donc s'écrire

$$x_i \leftarrow \frac{1}{A_{i,i}} (b_i - S).$$

Mais où calculer la valeur S? 4 choix possible : avant la ligne 1, entre les lignes 1 et 2, entre les lignes 2 et 3 ou après la ligne 3.
On ne peut pas calculer S après utilisation de sa valeur dans le calcul de x_i! ce qui élimine les 2 derniers choix. Ensuite, on ne peut sortir le calcul de S de la boucle puique la somme dépend de l'indice de boucle i : ce qui élimine le premier choix. On doit donc calculer S dans la boucle et avant le calcul de x_i.

Algorithme 4.2 $\mathcal{R}_1$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $\quad x_i \leftarrow \frac{1}{A_{i,i}} \left(b_i - \sum_{j=1}^{i-1} A_{i,j} x_j \right)$
3: **Fin Pour**

Algorithme 4.2 $\mathcal{R}_2$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $\quad S \leftarrow \sum_{j=1}^{i-1} A_{i,j} x_j$
3: $\quad x_i \leftarrow (b_i - S)/A_{i,i}$
4: **Fin Pour**

Maintenant que l'on a isolé la *difficulté*, il reste à détailler le calcul de S. Celui-ci se fait **intégralement** en lieu et place de la ligne 2.

Algorithme 4.2 $\mathcal{R}_3$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $\quad S \leftarrow \sum_{j=1}^{i-1} A_{i,j} x_j$
3: $\quad x_i \leftarrow (b_i - S)/A_{i,i}$
4: **Fin Pour**

Algorithme 4.2 $\mathcal{R}_4$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $\quad S \leftarrow 0$
3: $\quad$ **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
4: $\quad\quad S \leftarrow S + A(i,j) * x(j)$
5: $\quad$ **Fin Pour**
6: $\quad x_i \leftarrow (b_i - S)/A_{i,i}$
7: **Fin Pour**

Insister sur $S \leftarrow 0$ à l'intérieur de la boucle en i? (erreur trop courante des débutants)
On obtient alors l'algorithme final

Algorithme 4.2 Fonction RSLTriInf permettant de résoudre le système linéaire triangulaire inférieur inversible

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}.$$

{algo:RSL:TriInf

Données : $\mathbb{A}$: matrice triangulaire de $\mathcal{M}_n(\mathbb{R})$ inférieure inversible.
$\boldsymbol{b}$: vecteur de $\mathbb{R}^n$.
Résultat : $\boldsymbol{x}$: vecteur de $\mathbb{R}^n$.

1: **Fonction** $\boldsymbol{x} \leftarrow$ RSLTriInf($\mathbb{A}, \boldsymbol{b}$)
2: $\quad$ **Pour** $i \leftarrow 1$ **à** n **faire**
3: $\quad\quad S \leftarrow 0$
4: $\quad\quad$ **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
5: $\quad\quad\quad S \leftarrow S + A(i,j) * x(j)$
6: $\quad\quad$ **Fin Pour**
7: $\quad\quad x(i) \leftarrow (b(i) - S)/A(i,i)$
8: $\quad$ **Fin Pour**
9: **Fin Fonction**

Matrices triangulaires supérieures

Soit $\mathbb{A}$ une matrice de $\mathcal{M}_n(\mathbb{R})$ triangulaire supérieure inversible et $\boldsymbol{b} \in \mathbb{R}^n$. On veut résoudre le système linéaire

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \begin{pmatrix} A_{1,1} & \dots & \dots & A_{1,n} \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & A_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ \vdots \\ b_n \end{pmatrix}$$

On remarque que l'on peut calculer successivement $x_n, x_{n-1}, \dots, x_1$, car il est possible de calculer x_i si on connait $x_{i+1}, \dots, x_n$: c'est la **méthode de remontée**. En effet, on a

$$(\mathbb{A}\boldsymbol{x})_i = b_i, \ \forall i \in [\![1, n]\!].$$

et donc, par définition d'un produit matrice-vecteur,

$$\sum_{j=1}^{n} A_{i,j} x_j = b_i, \ \forall i \in [\![1, n]\!].$$

Comme $\mathbb{A}$ est une matrice triangulaire supérieure, on a (voir Définition A.37) $A_{i,j} = 0$ si $j < i$. Ceci donne alors pour tout $i \in [\![1, n]\!]$

$$\begin{aligned} b_i &= \sum_{j=1}^{i-1} \underbrace{A_{i,j}}_{=0} x_j + A_{i,i} x_i + \sum_{j=i+1}^{n} A_{i,j} x_j \\ &= A_{i,i} x_i + \sum_{j=i+1}^{n} A_{i,j} x_j \end{aligned}$$

De plus la matrice $\mathbb{A}$ étant triangulaire inversible ses éléments diagonaux sont tous non nuls. On obtient donc x_i en fonction des $x_{i+1}, \dots, x_n$:

L:MD:MP:TS:eq03}

$$x_i = \frac{1}{A_{i,i}} \left(b_i - \sum_{j=i+1}^{n} A_{i,j} x_j \right), \ \forall i \in [\![1, n]\!]. \tag{4.11}$$

Algorithme 4.3 $\mathcal{R}_0$

1: Résoudre $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$ en calculant successivement $x_n, \ x_{n-1}, \ \dots, x_1$.

Algorithme 4.3 $\mathcal{R}_1$

1: **Pour** $i \leftarrow n$ **à** 1 **faire**(**pas de** -1)
2: calculer x_i connaissant $x_{i+1}, \dots, x_n$ à l'aide de l'équation (??)
3: **Fin Pour**

Algorithme 4.3 $\mathcal{R}_1$

1: **Pour** $i \leftarrow n$ **à** 1 **faire**(**pas de** -1)
2: Calculer x_i connaissant $x_{i+1}, \dots, x_n$ à l'aide de l'équation (??)
3: **Fin Pour**

Algorithme 4.3 $\mathcal{R}_2$

1: **Pour** $i \leftarrow n$ **à** 1 **faire**(**pas de** -1)
2: $S \leftarrow \sum_{j=i+1}^{n} A_{i,j} x_j$
3: $x_i \leftarrow (b_i - S)/A_{i,i}$
4: **Fin Pour**

Algorithme 4.3 $\mathcal{R}_3$

1: **Pour** $i \leftarrow n$ **à** 1 **faire**(**pas de** -1)
2: $S \leftarrow \sum_{j=i+1}^{n} A_{i,j} x_j$
3: $x_i \leftarrow (b_i - S)/A_{i,i}$
4: **Fin Pour**

Algorithme 4.3 $\mathcal{R}_4$

1: **Pour** $i \leftarrow n$ **à** 1 **faire**(**pas de** -1)
2: $S \leftarrow 0$
3: **Pour** $j \leftarrow i + 1$ **à** n **faire**
4: $S \leftarrow S + A(i, j) * x(j)$
5: **Fin Pour**
6: $x_i \leftarrow (b_i - S)/A_{i,i}$
7: **Fin Pour**

On obtient alors l'algorithme final

Algorithme 4.3 Fonction RSLTriSup permettant de résoudre le système linéaire triangulaire supérieur inversible

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}.$$

SL:TriSup}

Données : $\mathbb{A}$: matrice triangulaire de $\mathcal{M}_n(\mathbb{R})$ supérieure inversible.
$\boldsymbol{b}$: vecteur de $\mathbb{R}^n$.
Résultat : $\boldsymbol{x}$: vecteur de $\mathbb{R}^n$.

```
1: Fonction x ← RSLTriSup( A, b )
2:     Pour i ← n à 1 faire(pas de −1)
3:         S ← 0
4:         Pour j ← i + 1 à n faire
5:             S ← S + A(i, j) ∗ x(j)
6:         Fin Pour
7:         x(i) ← (b(i) − S)/A(i, i)
8:     Fin Pour
9: Fin Fonction
```

4.2.2 Méthode de Gauss-Jordan, écriture matricielle

Soient $\mathbb{A} \in \mathcal{M}_{\mathbb{K}}()$ une matrice inversible et $\boldsymbol{b} \in \mathbb{K}^n$.
On va tout d'abord rappeller (très) brièvement l'**algorithme d'élimination** ou **algorithme de Gauss-Jordan** permettant de transformer le système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$ en un système linéaire équivalent dont la matrice est triangulaire supérieure. Ce dernier système se résoud par la méthode de remontée.
Ensuite, on va réécrire cet algorithme sous forme algébrique pour obtenir le théorème ...

Cette méthode doit son nom aux mathématiciens Carl Friedrich Gauss (1777-1855, mathématicien, astronome et physicien allemand) et Wilhelm Jordan (1842-1899, mathématicien et géodésien allemand) mais elle est connue des Chinois depuis au moins le Ier siècle de notre ère. Elle est référencée dans l'important livre chinois *Jiuzhang suanshu* ou *Les Neuf Chapitres sur l'art mathématique*, dont elle constitue le huitième chapitre, sous le titre « Fang cheng » (la disposition rectangulaire). La méthode est présentée au moyen de dix-huit exercices. Dans son commentaire daté de 263, Liu Hui en attribue la paternité à Chang Ts'ang, chancelier de l'empereur de Chine au IIe siècle avant notre ère.

Algorithme de Gauss-Jordan usuel

Pour la résolution du système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$ l'algorithme de Gauss-Jordan produit la forme échelonnée (réduite) d'une matrice à l'aide d'opérations élémentaires sur les lignes du système. Trois types d'opérations élémentaires sont utilisées:

- Echange de deux lignes ;
- Multiplication d'une ligne par un scalaire non nul ;
- Ajout du multiple d'une ligne à une autre ligne.

A l'aide de ces opérations élémentaires cet algorithme permet donc de transformer le système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$ en le système équivalent $\mathbb{U}\boldsymbol{x} = \boldsymbol{f}$ où $\mathbb{U}$ est triangulaire supérieure. En fait, l'algorithme va transformer la matrice $\mathbb{A}$ et le second membre $\boldsymbol{b}$ pour aboutir à un système dont la matrice est triangulaire supérieure.

Algorithme 4.4 Algorithme de Gauss-Jordan formel pour la résolution de $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

Pour $j \leftarrow 1$ **à** $n-1$ **faire**
  Rechercher l'indice k de la ligne du pivot (sur la colonne j, $k \in [\![j, n]\!]$)
  Permuter les lignes j ($\mathcal{L}_j$) et k ($\mathcal{L}_k$) du système si besoin.
  Pour $i \leftarrow j+1$ **à** n **faire**
    Eliminer en effectuant $\mathcal{L}_i \leftarrow \mathcal{L}_i - \frac{A_{i,j}}{A_{j,j}}\mathcal{L}_j$
  Fin Pour
Fin Pour
Résoudre le système triangulaire supérieur par la méthode de la remontée.

On va maintenant voir comment écrire cet algorithme de manière plus détaillée. Pour conserver sa lisibilité, on choisi pour chaque ligne un peu délicate de créer et d'utiliser une fonction dédiée à cette tâche.

{algo:RSL:GaussFormel}

Algorithme 4.5 Algorithme de Gauss-Jordan avec fonctions pour la résolution de $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{R})$ inversible.
$\boldsymbol{b}$: vecteur de $\mathbb{R}^n$.
Résultat : $\boldsymbol{x}$: vecteur de $\mathbb{R}^n$.

Fonction $\boldsymbol{x} \leftarrow$ RSLGauss($\mathbb{A}, \boldsymbol{b}$)
  Pour $j \leftarrow 1$ **à** $n-1$ **faire**
    $k \leftarrow$ ChercheIndPivot$(\mathbb{A}, j)$ ▷ ChercheIndPivot à écrire
    $[\mathbb{A}, \boldsymbol{b}] \leftarrow$ PermLignesSys$(\mathbb{A}, \boldsymbol{b}, j, k)$ ▷ PermLignesSys à écrire
    Pour $i \leftarrow j+1$ **à** n **faire**
      $[\mathbb{A}, \boldsymbol{b}] \leftarrow$ CombLignesSys$(\mathbb{A}, \boldsymbol{b}, j, i, -A(i,j)/A(j,j))$ ▷ CombLignesSys à écrire
    Fin Pour
  Fin Pour
  $\boldsymbol{x} \leftarrow$ RSLTriSup$(\mathbb{A}, \boldsymbol{b})$ ▷ RSLTriSup déjà écrite
Fin Fonction

Bien évidemment, il reste à décrire et écrire les différentes fonctions utilisées dans cette fonction :

Fonction $k \leftarrow$ **ChercheIndPivot**$(\mathbb{A}, j)$ **:** recherche $k \in [\![j, n]\!]$ tel que $\forall l \in [\![j, n]\!]$, $|A_{l,j}| \leqslant |A_{k,j}|$.

Fonction $[\mathbb{A}, \boldsymbol{b}] \leftarrow$ **PermLignesSys**$(\mathbb{A}, \boldsymbol{b}, i, k)$ **:** permute les lignes i et k de la matrice $\mathbb{A}$ ainsi que celles du vecteur $\boldsymbol{b}$.

Fonction $[\mathbb{A}, \boldsymbol{b}] \leftarrow$ **CombLignesSys**$(\mathbb{A}, \boldsymbol{b}, j, i, \alpha)$ **:** remplace la ligne i de la matrice A par la combinaison linéaire $\mathcal{L}_i \leftarrow \mathcal{L}_i + \alpha\mathcal{L}_j$. De même on remplace la ligne i de $\boldsymbol{b}$ par $b_i + \alpha b_j$.

Ces trois fonctions sont simples à écrire et sont données en Algorithmes 4.6, 4.7 et 4.8.

{algo:RSL:ChercheIndPivot}

Algorithme 4.6 Recherche d'un pivot pour l'algorithme de Gauss-Jordan.
Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{R})$.
j : entier, $1 \leqslant j \leqslant n$.
Résultat : k : entier, indice ligne pivot

Fonction $k \leftarrow$ ChercheIndPivot($\mathbb{A}, j$)
  $k \leftarrow j$, pivot $\leftarrow |\mathbb{A}(j,j)|$
  Pour $i \leftarrow j+1$ **à** n **faire**
    Si $|\mathbb{A}(i,j)| >$ pivot **alors**
      $k \leftarrow i$, pivot $\leftarrow |\mathbb{A}(i,j)|$
    Fin Si
  Fin Pour
Fin Fonction

{algo:RSL:PermLignesSys}

Algorithme 4.7 Permutte deux lignes d'une matrice et d'un vecteur.
Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$.
$\boldsymbol{b}$: vecteur de $\mathbb{K}^n$.
j, k : entiers, $1 \leqslant j, k \leqslant n$.
Résultat : $\mathbb{A}$ et $\boldsymbol{b}$ modifiés.

Fonction $[\mathbb{A}, \boldsymbol{b}] \leftarrow$ PermLignesSys($\mathbb{A}, \boldsymbol{b}, j, k$)
  Pour $l \leftarrow 1$ **à** n **faire**
    $t \leftarrow \mathbb{A}(l,j)$, $\mathbb{A}(l,j) \leftarrow \mathbb{A}(l,k)$, $\mathbb{A}(l,k) \leftarrow t$
  Fin Pour
  $t \leftarrow \boldsymbol{b}(j)$, $\boldsymbol{b}(j) \leftarrow \boldsymbol{b}(k)$, $\boldsymbol{b}(k) \leftarrow t$
Fin Fonction

{algo:RSL:CombLignesSys}

Algorithme 4.8 Combinaison linéaire $\mathcal{L}_i \leftarrow \mathcal{L}_i + \alpha\mathcal{L}_j$ appliqué à une matrice et à un vecteur.
Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$.
$\boldsymbol{b}$: vecteur de $\mathbb{K}^n$.
j, i : entiers, $1 \leqslant j, i \leqslant n$.
alpha : scalaire de $\mathbb{K}$
Résultat : $\mathbb{A}$ et $\boldsymbol{b}$ modifiés.

Fonction $[\mathbb{A}, \boldsymbol{b}] \leftarrow$ CombLignesSys($\mathbb{A}, \boldsymbol{b}, j, i,$ alpha)
  Pour $k \leftarrow 1$ **à** n **faire**
    $\mathbb{A}(i,k) \leftarrow \mathbb{A}(i,k) +$ alpha$\mathbb{A}(j,k)$
  Fin Pour
  $\boldsymbol{b}(i) \leftarrow \boldsymbol{b}(i) +$ alpha$\boldsymbol{b}(j)$
Fin Fonction

Ecriture algébrique

Sous forme d'exercice :

Exercice 4.2.2

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ inversible.

Q. 1 *Montrer qu'il existe une matrice $\mathbb{G} \in \mathcal{M}_n(\mathbb{C})$ telle que $|\det(\mathbb{G})| = 1$ et $\mathbb{G}\mathbb{A}\boldsymbol{e}_1 = \alpha \boldsymbol{e}_1$ avec $\alpha \neq 0$ et $\boldsymbol{e}_1$ premier vecteur de la base canonique de C^n.*

Q. 2 *1. Montrer par récurrence sur l'ordre des matrices que pour toute matrice $\mathbb{A}_n \in \mathcal{M}_n(\mathbb{C})$ inversible, il existe une matrice $\mathbb{S}_n \in \mathcal{M}_n(\mathbb{C})$ telle que $|\det \mathbb{S}_n| = 1$ et $\mathbb{S}_n\mathbb{A}_n = \mathbb{U}_n$ avec $\mathbb{U}_n$ matrice triangulaire supérieure inversible.*

2. Soit $\boldsymbol{b} \in \mathbb{C}^n$. En supposant connue la décompostion précédente $\mathbb{S}_n\mathbb{A}_n = \mathbb{U}_n$, expliquer comment résoudre le système $\mathbb{A}_n\boldsymbol{x} = \boldsymbol{b}$.

Q. 3 *Que peut-on dire si $\mathbb{A}$ est non inversible?*

Indication : utiliser les résultats des exercices 4.1.2 et 4.1.3.

Correction Exercice 4.2.2

Q. 1 D'après le Lemme 4.2, si $A_{1,1} \neq 0$ le résultat est immédiat. Dans l'énoncé rien ne vient corroborer cette hypothèse. Toutefois, comme la matrice $\mathbb{A}$ est inversible, il existe au moins un $p \in [\![1, n]\!]$ tel que $A_{p,1} \neq 0$. On peut même choisir le premier indice p tel que $|A_{p,1}| = \max_{i\in[\![1,n]\!]} |A_{p,1}| > 0$ (pivot de l'algorithme de Gauss-Jordan). On note $\mathbb{P} = \mathbb{P}_n^{[1,p]}$ la matrice de permutation des lignes 1 et p (voir exercice 4.1.3, page 72). De plus on a

$$|\det \mathbb{P}| = 1 \quad \text{et} \quad \mathbb{P}^{-1} = \mathbb{P}.$$

Par construction $(\mathbb{P}\mathbb{A})_{1,1} = A_{p,1} \neq 0$, et on peut alors appliquer le Lemme 4.2 à la matrice $(\mathbb{P}\mathbb{A})$ pour obtenir l'existence d'une matrice $\mathbb{E} \in \mathcal{M}_n(\mathbb{C})$ vérifiant $\det \mathbb{E} = 1$ et telle que

$$\mathbb{E}(\mathbb{P}\mathbb{A})\boldsymbol{e}_1 = A_{p,1}\boldsymbol{e}_1.$$

En posant $\mathbb{G} = \mathbb{E}\mathbb{P}$ et $\alpha = A_{p,1}$, on obtient bien $\mathbb{G}\mathbb{A}\boldsymbol{e}_1 = \alpha\boldsymbol{e}_1$. De plus, on a

$$|\det \mathbb{G}| = |\det(\mathbb{E}\mathbb{P})| = |\det \mathbb{E} \times \det \mathbb{P}| = 1.$$

Remarque 4.3 La matrice $\mathbb{G}$ étant inversible, on a

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \mathbb{G}\mathbb{A}\boldsymbol{x} = \mathbb{G}\boldsymbol{b}$$

ce qui correspond à la première *permutation/élimination* de l'algorithme de Gauss-Jordan.

Q. 2 1. On veut démontrer par récurrence la propriété $(\mathcal{P}_n)$,

$$(\mathcal{P}_n) \quad \begin{cases} \forall \mathbb{A}_n \in \mathcal{M}_n(\mathbb{C}), \text{ inversible } \exists \mathbb{S}_n \in \mathcal{M}_n(\mathbb{C}), \ |\det \mathbb{S}_n| = 1, \text{ tel que} \\ \text{la matrice } \mathbb{U}_n = \mathbb{S}_n\mathbb{A} \text{ soit une triangulaire supérieure inversible} \end{cases}$$

Initialisation : Pour $n = 2$. Soit $\mathbb{A}_2 \in \mathcal{M}_2(\mathbb{C})$ inversible. En utilisant la question précédente il existe $\mathbb{G}_2 \in \mathcal{M}_2(\mathbb{C})$ telle que $|\det \mathbb{G}_2| = 1$ et $\mathbb{G}_2\mathbb{A}_2\boldsymbol{e}_1 = \alpha\boldsymbol{e}_1$ avec $\alpha \neq 0$ et $\boldsymbol{e}_1$ premier vecteur de la base canonique de $\mathbb{C}^2$. On note $\mathbb{U}_2 = \mathbb{G}_2\mathbb{A}_2$. Cette matrice s'écrit donc sous la forme

$$\mathbb{U}_2 = \begin{pmatrix} \alpha & \bullet \\ 0 & \bullet \end{pmatrix}$$

et elle est triangulaire supérieure. Les matrices $\mathbb{G}_2$ et $\mathbb{A}_2$ étant inversible, leur produit $\mathbb{U}_2$ l'est aussi. La proposition $(\mathcal{P}_2)$ est donc vérifiée avec $\mathbb{S}_2 = \mathbb{G}_2$.

Hérédité : Soit $n \geqslant 3$. On suppose que $(\mathcal{P}_{n-1})$ est vraie. Montrons que $(\mathcal{P}_n)$ est vérifiée.
Soit $\mathbb{A}_n \in \mathcal{M}_n(\mathbb{C})$ inversible. En utilisant la question précédente il existe $\mathbb{G}_n \in \mathcal{M}_n(\mathbb{C})$ telle que $|\det \mathbb{G}_n| = 1$ et $\mathbb{G}_n\mathbb{A}_n\boldsymbol{e}_1 = \alpha_n\boldsymbol{e}_1$ avec $\alpha_n \neq 0$ et $\boldsymbol{e}_1$ premier vecteur de la base canonique de $\mathbb{C}^n$. On note $\mathbb{V}_n = \mathbb{G}_n\mathbb{A}_n$. Cette matrice s'écrit donc sous la forme

$$\mathbb{V}_n = \left(\begin{array}{c|ccc} \alpha_n & \bullet & \dots & \bullet \\ \hline 0 & \bullet & \dots & \bullet \\ \vdots & \vdots & & \vdots \\ 0 & \bullet & \dots & \bullet \end{array}\right) \overset{\text{def}}{=} \left(\begin{array}{c|c} \alpha_n & \boldsymbol{c}_{n-1}^* \\ \hline 0 & \\ \vdots & \mathbb{B}_{n-1} \\ 0 & \end{array}\right)$$

où $\boldsymbol{c}_{n-1} \in \mathbb{C}^{n-1}$ et $\mathbb{B}_{n-1} \in \mathcal{M}_{n-1}(\mathbb{C})$. Comme $\mathbb{G}_n$ et $\mathbb{A}_n$ sont inversibles, $\mathbb{V}_n$ l'est aussi. On en déduit donc que $\mathbb{B}_{n-1}$ est inversible car $0 \neq \det \mathbb{V}_n = \alpha_n \times \det \mathbb{B}_{n-1}$ et $\alpha_n \neq 0$.

On peut donc utiliser la propriété $(\mathcal{P}_{n-1})$ (hyp. de récurrence) sur la matrice $\mathbb{B}_{n-1}$: il existe donc $\mathbb{S}_{n-1} \in \mathcal{M}_{n-1}(\mathbb{C})$, avec $|\det \mathbb{S}_{n-1}| = 1$, tel que la matrice $\mathbb{U}_{n-1} = \mathbb{S}_{n-1}\mathbb{B}_{n-1}$ soit une triangulaire supérieure inversible.

Soit $\mathbb{Q}_n \in \mathcal{M}_n(\mathbb{C})$ la matrice définie par

$$\mathbb{Q}_n = \left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline 0 & & & \\ \vdots & & \mathbb{S}_{n-1} & \\ 0 & & & \end{array}\right)$$

On a alors

$$\mathbb{Q}_n\mathbb{G}_n\mathbb{A}_n = \mathbb{Q}_n\mathbb{V}_n = \left(\begin{array}{c|ccc} 1 & 0 & \dots & 0 \\ \hline 0 & & & \\ \vdots & & \mathbb{S}_{n-1} & \\ 0 & & & \end{array}\right)\left(\begin{array}{c|c} \alpha_n & \boldsymbol{c}^*_{n-1} \\ \hline 0 & \\ \vdots & \mathbb{B}_{n-1} \\ 0 & \end{array}\right)$$

$$= \left(\begin{array}{c|c} \alpha_n & \boldsymbol{c}^*_{n-1} \\ \hline 0 & \\ \vdots & \mathbb{S}_{n-1}\mathbb{B}_{n-1} \\ 0 & \end{array}\right) = \left(\begin{array}{c|c} \alpha_n & \boldsymbol{c}^*_{n-1} \\ \hline 0 & \\ \vdots & \mathbb{U}_{n-1} \\ 0 & \end{array}\right) \stackrel{\text{def}}{=} \mathbb{U}_n$$

La matrice $\mathbb{U}_n$ est triangulaire supérieure inversible car $\mathbb{U}_{n-1}$ l'est aussi et $\alpha_n \neq 0$.

On pose $\mathbb{S}_n = \mathbb{Q}_n\mathbb{G}_n$. On a donc $\mathbb{S}_n\mathbb{A}_n = \mathbb{U}_n$ et comme $|\det \mathbb{S}_n| = 1$ car $|\det \mathbb{G}_n| = 1$ et $\det \mathbb{G}_n = 1$, ceci prouve la véracité de la proposition $(\mathcal{P}_n)$.

2. Comme $\mathbb{S}_n$ est inversible, on a en multipliant à gauche le système par $\mathbb{S}_n$

$$\mathbb{A}_n\boldsymbol{x} = \boldsymbol{b} \iff \mathbb{S}_n\mathbb{A}_n\boldsymbol{x} = \mathbb{S}_n\boldsymbol{b} \iff \mathbb{U}_n\boldsymbol{x} = \mathbb{S}_n\boldsymbol{b}$$

Pour déterminer le vecteur $\boldsymbol{x}$, on peut alors résoudre le dernier système par l'algorithme de remontée.

Q. 3 (rapide) Si $\mathbb{A}$ est non inversible, alors dans la première question nous ne sommes pas assurés d'avoir $\alpha \neq 0$. Cependant l'existence de la matrice $\mathbb{G}$ reste avérée.

Pour la deuxième question, le seule changement vient du fait que la matrice $\mathbb{U}_n$ n'est plus inversible.

◇

On a donc démontré le théorème suivant

Théorème 4.4

Soit $\mathbb{A}$ une matrice carrée, inversible ou non. Il existe (au moins) une matrice inversible $\mathbb{G}$ telle que $\mathbb{G}\mathbb{A}$ soit triangulaire supérieure.

Proof. voir Exercice 4.2.2, page 79. □

4.2.3 Factorisation $\mathbb{L}\mathbb{U}$

Avant de citer le théorème principal, on va faire un "petit" exercice...

Exercice 4.2.3: ★★★★★

{MD:LU:exo:01}

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice dont les sous-matrices principales d'ordre i, notées Δ_i, $i \in [\![1,n]\!]$ (voir Definition A.41, page 121) sont inversibles.
Montrer qu'il existe des matrices $\mathbb{E}^{[k]} \in \mathcal{M}_n(\mathbb{C})$, $k \in [\![1,n-1]\!]$, triangulaires inférieures à diagonale unité telles que la matrice $\mathbb{U}$ définie par

$$\mathbb{U} = \mathbb{E}^{[n-1]} \dots \mathbb{E}^{[1]} \mathbb{A}$$

soit triangulaire supérieure avec $U_{i,i} = \det \Delta_i / (U_{1,1} \times \cdots \times U_{i-1,i-1})$, $\forall i \in [\![1,n]\!]$.

Correction Exercice 4.2.3

On note $\mathbb{A}^{[0]} = \mathbb{A}$. On va démontrer par récurrence finie sur $k \in [\![1,n-1]\!]$, qu'il existe une matrice $\mathbb{E}^{[k]} \in \mathcal{M}_n(\mathbb{C})$, tel que la matrice $\mathbb{A}^{[k]}$ définie itérativement par

$$\mathbb{A}^{[k]} = \mathbb{E}^{[k]} \mathbb{A}^{[k-1]}$$

s'écrive sous la forme bloc

$$\mathbb{A}^{[k]} = \left(\begin{array}{cccc:ccc} \alpha_1 & \bullet & \cdots & \bullet & \bullet & \cdots & \bullet \\ 0 & \ddots & \ddots & \vdots & \vdots & & \vdots \\ \vdots & \ddots & \ddots & \bullet & \vdots & & \vdots \\ 0 & \dots & 0 & \alpha_k & \bullet & \cdots & \bullet \\ \hdashline 0 & \dots & \dots & 0 & \bullet & \cdots & \bullet \\ \vdots & \ddots & \ddots & \vdots & \vdots & & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & & \vdots \\ 0 & \dots & \dots & 0 & \bullet & \cdots & \bullet \end{array}\right) \tag{4.12}$$

{RSL:MD:exo10:eq}

avec $\alpha_1 = A_{1,1}$ et $\forall i \in [\![2,k]\!]$, $\alpha_i = \det \Delta_i / (\alpha_1 \times \cdots \times \alpha_{i-1})$.

Initialisation ($k = 1$)**:** On a $A_{1,1} \neq 0$ car $\Delta_1 = A_{1,1}$ et $\det \Delta_1 \neq 0$. D'après le Lemme 4.2, il existe une matrice $\mathbb{E}^{[1]} \in \mathcal{M}_n(\mathbb{C})$, triangulaire inférieure à diagonale unité telle que $\mathbb{E}^{[1]} \mathbb{A} \boldsymbol{e}_1 = A_{1,1} \boldsymbol{e}_1$ où $\boldsymbol{e}_1$ est le premier vecteur de la base canonique de $\mathbb{C}^n$. On a alors

$$\mathbb{A}^{[1]} = \mathbb{E}^{[1]} \mathbb{A} = \left(\begin{array}{c:ccc} \alpha_1 & \bullet & \cdots & \bullet \\ \hdashline 0 & \bullet & \cdots & \bullet \\ \vdots & \vdots & & \vdots \\ 0 & \bullet & \cdots & \bullet \end{array}\right)$$

avec $\alpha_1 = A_{1,1} = \det \Delta_1$.

Hérédité ($k < n-1$)**:** Supposons construite la matrice $\mathbb{A}^{[k]}$. Il existe donc k matrices, $\mathbb{E}^{[1]}, \dots, \mathbb{E}^{[k]}$, triangulaires inférieures à diagonale unité telles que

$$\mathbb{A}^{[k]} = \mathbb{E}^{[k]} \dots \mathbb{E}^{[1]} \mathbb{A}.$$

- On va montrer que $\alpha_{k+1} \stackrel{\text{def}}{=} A^{[k]}_{k+1,k+1} \neq 0$. Pour celà, on réécrit la matrice $\mathbb{A}^{[k]}$ sous forme bloc, avec comme premier bloc diagonale le bloc de dimension $k+1$:

$$\mathrm{A}^{[k]} = \left(\begin{array}{ccccc:ccc} \alpha_1 & \bullet & \cdots & \bullet & \bullet & \bullet & \cdots & \bullet \\ 0 & \ddots & \ddots & \vdots & \bullet & \vdots & & \vdots \\ \vdots & \ddots & \ddots & \bullet & \bullet & \vdots & & \vdots \\ 0 & \dots & 0 & \alpha_k & \bullet & \vdots & \cdots & \vdots \\ 0 & \dots & \dots & 0 & \alpha_{k+1} & \bullet & \cdots & \bullet \\ \hdashline 0 & \cdots & \cdots & 0 & \bullet & \bullet & \cdots & \bullet \\ \vdots & & & \vdots & \bullet & \vdots & & \vdots \\ 0 & \dots & \dots & 0 & \bullet & \bullet & \cdots & \bullet \end{array}\right)$$

$k+1$

La matrice $\mathbb{G}^{[k]} \overset{\text{def}}{=} \mathbb{E}^{[k]} \dots \mathbb{E}^{[1]}$ est triangulaire inférieure à diagonale unité car produit de matrices triangulaires inférieures à diagonale unité (voir ...). Le produit de $\mathbb{G}^{[k]}\mathbb{A}$ s'écrit alors sous forme bloc

$$\mathrm{G}^{[k]}\mathbb{A} = \left(\begin{array}{cccc:ccccc} 1 & 0 & \cdots & 0 & 0 & \cdots & \cdots & 0 \\ \bullet & \ddots & \ddots & \vdots & \vdots & & & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots & & & \vdots \\ \bullet & \dots & \bullet & 1 & 0 & \cdots & \cdots & 0 \\ \hdashline \bullet & \dots & \dots & \bullet & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots & \bullet & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & 0 \\ \bullet & \dots & \dots & \bullet & \bullet & \cdots & \bullet & 1 \end{array}\right) \left(\begin{array}{ccc:ccc} & & & \bullet & \cdots & \bullet \\ & \Delta_{k+1} & & \vdots & & \vdots \\ & & & \bullet & \cdots & \bullet \\ \hdashline \bullet & \cdots & \bullet & \bullet & \cdots & \bullet \\ \vdots & & \vdots & \vdots & & \vdots \\ \bullet & \cdots & \bullet & \bullet & \cdots & \bullet \end{array}\right)$$

Comme $\mathbb{A}^{[k]} = \mathbb{G}^{[k]}\mathbb{A}$, en utilisant les règles de multiplication par blocs des matrices on obtient

$$\begin{pmatrix} \alpha_1 & \bullet & \cdots & \bullet & \bullet \\ 0 & \ddots & \ddots & \vdots & \bullet \\ \vdots & \ddots & \ddots & \bullet & \bullet \\ 0 & \dots & 0 & \alpha_k & \bullet \\ 0 & \dots & \dots & 0 & \alpha_{k+1} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ \bullet & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ \bullet & \dots & \bullet & 1 \end{pmatrix} \begin{pmatrix} & & & \\ & \Delta_{k+1} & & \\ & & & \end{pmatrix}$$

En prenant le déterminant de cette dernière équation, et en utilisant le fait que le determinant de matrices triangulaires est le produit de ses coefficients diagonaux, on obtient

$$\prod_{i=1}^{k+1} \alpha_i = \det \Delta_{k+1}.$$

Or Δ_{k+1} inversible par hypothèse, ce qui entraine $\det \Delta_{k+1} \neq 0$ et donc $\alpha_i \neq 0$, $\forall i \in [\![1, k+1]\!]$ et on a

$$\alpha_{k+1} = \frac{\det \Delta_{k+1}}{\prod_{i=1}^{k} \alpha_i} \neq 0.$$

- Montrons l'existence d'une matrice triangulaire inférieure à diagonale unité permettant d'éliminer les termes sous diagonaux de la colonne $k+1$ de $\mathbb{A}^{[k]}$.
  Revenons à l'écriture bloc de premier bloc diagonal de dimension k. On a

$$\mathbb{A}^{[k]} = \left(\begin{array}{cccc:ccccc} \alpha_1 & \bullet & \cdots & \bullet & \bullet & \cdots & \cdots & \bullet \\ 0 & \ddots & \ddots & \vdots & \vdots & & & \vdots \\ \vdots & \ddots & \ddots & \bullet & \vdots & & & \vdots \\ 0 & \dots & 0 & \alpha_k & \bullet & \cdots & \cdots & \bullet \\ \hdashline 0 & \dots & \dots & 0 & \alpha_{k+1} & \bullet & \cdots & \bullet \\ \vdots & \ddots & \ddots & \vdots & \bullet & \cdots & \cdots & \bullet \\ \vdots & \ddots & \ddots & \vdots & \vdots & & & \vdots \\ 0 & \dots & \dots & 0 & \bullet & \cdots & \cdots & \bullet \end{array}\right) \overset{\text{def}}{=} \left(\begin{array}{c:c} \mathbb{U}^{[k]} & \mathbb{F}^{[k]} \\ \hdashline \mathbb{O} & \mathbb{V}^{[k]} \end{array}\right)$$

Nous sommes exactement dans le cas de figure étudié dans l'exercice 4.1.2. En effet, avec les notations de cet exercice et si l'on pose $\boldsymbol{v} = \mathbb{V}^{[k]}_{:,1} = (A^{[k]}_{k+1,k+1}, \dots, A^{[k]}_{n,k+1})^{\mathrm{t}} \in \mathbb{C}^{n-(k+1)}$ on a alors $v_1 = A^{[k]}_{k+1,k+1} = \alpha_{k+1} \neq 0$ et l'on peut définir la matrice $\mathbb{E}^{[k+1]}$, triangulaire inférieure à diagonale unité, par

$$\mathbb{E}^{[k+1]} = \mathbb{E}^{[k,\boldsymbol{v}]} \overset{\text{def}}{=} \left(\begin{array}{c:c} \mathbb{I}_k & \mathbb{O} \\ \hdashline \mathbb{O} & \mathbb{E}^{[\boldsymbol{v}]} \end{array}\right) \in \mathcal{M}_n(\mathbb{C})$$

avec $\mathbb{E}^{[\boldsymbol{v}]} \in \mathcal{M}_{n-k}(\mathbb{C})$ triangulaire inférieure à diagonale unité (définie dans l'exercice 4.1.2) telle que

$$\mathbb{E}^{[\boldsymbol{v}]}\mathbb{V}^{[k]} = \begin{pmatrix} \alpha_{k+1} & \bullet & \cdots & \bullet \\ 0 & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ 0 & \bullet & \cdots & \bullet \end{pmatrix}$$

On a alors

$$\begin{aligned} \mathbb{A}^{[k+1]} \overset{\text{def}}{=} \mathbb{E}^{[k+1]}\mathbb{A}^{[k]} &= \left(\begin{array}{c|c} \mathbb{I}_k & \mathbb{O} \\ \hline \mathbb{O} & \mathbb{E}^{[\boldsymbol{v}]} \end{array}\right)\left(\begin{array}{c|c} \mathbb{U}^{[k]} & \mathbb{F}^{[k]} \\ \hline \mathbb{O} & \mathbb{V}^{[k]} \end{array}\right) \\ &= \left(\begin{array}{c|c} \mathbb{U}^{[k]} & \mathbb{F}^{[k]} \\ \hline \mathbb{O} & \mathbb{E}^{[\boldsymbol{v}]}\mathbb{V}^{[k]} \end{array}\right) \end{aligned}$$

et donc $\mathbb{A}^{[k+1]}$ s'écrit bien sous la forme (4.12) au rang $k+1$.

Final ($k = n-1$)**:** On a donc

$$\mathbb{U} = \mathbb{A}^{[n-1]} \overset{\text{def}}{=} \mathbb{E}^{[n-1]} \times \cdots \times \mathbb{E}^{[1]} \times \mathbb{A} = \left(\begin{array}{cccc|c} \alpha_1 & \bullet & \cdots & \cdots & \bullet \\ 0 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \alpha_{n-1} & \bullet \\ \hline 0 & \cdots & \cdots & 0 & \bullet \end{array}\right) \tag{4.13}$$

{RSL:MD:exo10:eq

où pour tout $k \in [\![1, n-1]\!]$ les matrices $\mathbb{E}^{[k]}$ sont triangulaires inférieures à diagonale unité.

Pour achever l'exercice, il reste à démontrer que

$$U_{n,n} = \det \Delta_n / (U_{1,1} \times \cdots \times U_{n-1,n-1}).$$

En effet, en prenant le déterminant dans (4.13) on obtient

$$\det\left(\mathbb{E}^{[n-1]} \times \cdots \times \mathbb{E}^{[1]} \times \mathbb{A}\right) = \det \left(\begin{array}{cccc|c} U_{1,1} & \bullet & \cdots & \cdots & \bullet \\ 0 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & U_{n-1,n-1} & \bullet \\ \hline 0 & \cdots & \cdots & 0 & U_{n,n} \end{array}\right)$$

Comme le déterminant d'un produit de matrices est égale au produit des déterminants des matrices on a

$$\begin{aligned} \det\left(\mathbb{E}^{[n-1]} \times \cdots \times \mathbb{E}^{[1]} \times \mathbb{A}\right) &= \det \mathbb{E}^{[n-1]} \times \cdots \times \det \mathbb{E}^{[1]} \times \det \mathbb{A} \\ &= \det \mathbb{A} \end{aligned}$$

car les matrices $\mathbb{E}^{[k]}$ sont triangulaires inférieures à diagonale unité et donc $\det \mathbb{E}^{[k]} = 1$, $\forall k \in [\![1, n-1]\!]$. De plus, le déterminant d'une matrice traingulaire supérieure est égale au produit de ses coefficients diagonaux et donc

$$\det \left(\begin{array}{cccc|c} U_{1,1} & \bullet & \cdots & \cdots & \bullet \\ 0 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & U_{n-1,n-1} & \bullet \\ \hline 0 & \cdots & \cdots & 0 & U_{n,n} \end{array}\right) = U_{n,n} \prod_{k=1}^{n-1} Uk, k.$$

On a alors

$$\det \mathbb{A} = \det \Delta_n = U_{n,n} \prod_{k=1}^{n-1} Uk, k \neq 0.$$

◇

Théorème 4.5: Factorisation $\mathbb{LU}$ ★★★★★

{labRSL:MD:th

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice dont les sous-matrices principales sont inversibles alors il existe une unique matrice $\mathbb{L} \in \mathcal{M}_n(\mathbb{C})$ triangulaire inférieure (*lower triangular* en anglais) à diagonale unité et une unique matrice $\mathbb{U} \in \mathcal{M}_n(\mathbb{C})$ triangulaire supérieure (*upper triangular* en anglais) telles ques

$$\mathbb{A} = \mathbb{LU}.$$

Proof. En utilisant le résultat de l'exercice 4.2.3, il existe des matrices $\mathbb{E}^{[k]} \in \mathcal{M}_n(\mathbb{C})$, $k \in [\![1, n-1]\!]$, triangulaires inférieures à diagonale unité telles que la matrice $\mathbb{U}$ définie par

$$\mathbb{U} = \mathbb{E}^{[n-1]} \dots \mathbb{E}^{[1]} \mathbb{A}$$

soit triangulaire supérieure avec $U_{i,i} \neq 0$, $\forall i \in [\![1, n]\!]$. On pose $\mathbb{G} = \mathbb{E}^{[n-1]} \dots \mathbb{E}^{[1]}$. La matrice $\mathbb{G}$ est donc aussi triangulaire inférieure à diagonale unité. Elle est donc inversible est son inverse est triangulaire inférieure à diagonale unité (voir Proposition A.39, page 120). En notant $\mathbb{L} = \mathbb{G}^{-1}$ on a $\mathbb{A} = \mathbb{LU}$.

On vient de démontrer l'existence d'une factorisation $\mathbb{LU}$ de la matrice $\mathbb{A}$. Pour démontrer l'unicité, on va supposer qu'il existe deux factorisations $\mathbb{LU}$ de $\mathbb{A}$ i.e.

$$\mathbb{A} = \mathbb{L}_1\mathbb{U}_1 = \mathbb{L}_2\mathbb{U}_2.$$

avec $\mathbb{L}_1, \mathbb{L}_2$ matrices triangulaires inférieures à diagonale unité et $\mathbb{U}_1, \mathbb{U}_2$ matrices triangulaires supérieures (inversibles). En multipliant l'équation $\mathbb{L}_1\mathbb{U}_1 = \mathbb{L}_2\mathbb{U}_2$ à gauche par $\mathbb{L}_1^{-1}$ et à droite par $\mathbb{U}_2^{-1}$ on obtient

{RSL:MD:LU:eq10}

$$\mathbb{U}_1\mathbb{U}_2^{-1} = \mathbb{L}_1^{-1}\mathbb{L}_2. \tag{4.14}$$

La matrice $\mathbb{L}_1^{-1}\mathbb{L}_2$ est triangulaire inférieure à diagonale unité car prduit de deux matrices triangulaires inférieures à diagonale unité. Elle est égale à la matrice $\mathbb{U}_1\mathbb{U}_2^{-1}$ qui elle est triangulaire supérieure (car prduit de deux matrices triangulaires supérieures). Donc $\mathbb{L}_1^{-1}\mathbb{L}_2$ est à la fois une matrice triangulaire supérieure et inférieure : elle est donc diagonale. Comme elle est à diagonale unité on en déduit que $\mathbb{L}_1^{-1}\mathbb{L}_2 = \mathbb{I}$ et donc $\mathbb{L}_1 = \mathbb{L}_2$. De l'équation (4.14), on tire alors $\mathbb{U}_1 = \mathbb{U}_2$. □

Remarque 4.6 Si la matrice $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ est inversible mais que ses sous-matrices principales ne sont pas toutes inversibles, il est possible par des permutations préalables de lignes de la matrice de se ramener à une matrice telle que ses sous-matrices principales soient inversibles.

Théorème 4.7: Factorisation $\mathbb{LU}$ avec permutations ★★★★☆

:th:factLUavecP}

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice inversible. Il existe une matrice $\mathbb{P}$, produit de matrices de permutation, une matrice $\mathbb{L} \in \mathcal{M}_n(\mathbb{C})$ triangulaire inférieure à diagonale unité et une matrice $\mathbb{U} \in \mathcal{M}_n(\mathbb{C})$ triangulaire supérieure telles ques

$$\mathbb{PA} = \mathbb{LU}. \tag{4.15}$$

Corollaire 4.8: ★★★☆☆

:coro:factLUDefPos}

Si $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ est une matrice hermitienne définie positive alors elle admet une unique factorisation $\mathbb{LU}$.

Proof. (*indication*) Si la matrice $\mathbb{A}$ est hermitienne définie positive alors toutes ses sous-matrices principales sont définies positives et donc inversibles. □

Résolution d'un système linéaire par factorisation $\mathbb{LU}$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{R})$ une matrice dont les sous-matrices principales sont inversibles et $\boldsymbol{b} \in \mathbb{R}^n$. On veut résoudre le système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$. Pour celà, grâce au théorème 4.5, on obtient :

{ExLU:eq-1}

Trouver $\boldsymbol{x} \in \mathbb{R}^n$ tel que

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}. \tag{4.16}$$

est équivalent à

Trouver $\boldsymbol{x} \in \mathbb{R}^n$ solution de

$$\mathbb{U}\boldsymbol{x} = \boldsymbol{y} \tag{4.17}$$ {ExLU:eq-2}

avec $\boldsymbol{y} \in \mathbb{R}^n$ solution de

$$\mathbb{L}\boldsymbol{y} = \boldsymbol{b}. \tag{4.18}$$ {ExLU:eq-3}

Ceci permet donc de découper le problème initial en trois sous-problèmes plus simples. De plus, ceux-ci peuvent se traiter de manière indépendante. On obtient alors l'algorithme suivant

Algorithme 4.9 Fonction RSLFactLU permettant de résoudre, par une factorisation $\mathbb{LU}$, le système linéaire

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$$

où $\mathbb{A}$ une matrice de $\mathcal{M}_n(\mathbb{R})$ définie positive et $\boldsymbol{b} \in \mathbb{R}^n$.

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{R})$ dont les sous-matrices principales sont inversibles définie positive,
$\boldsymbol{b}$: vecteur de $\mathbb{R}^n$.
Résultat : $\boldsymbol{x}$: vecteur de $\mathbb{R}^n$.

1: **Fonction** $\boldsymbol{x} \leftarrow$ RSLFactLU($\mathbb{A}, \boldsymbol{b}$)
2: $[\mathbb{L}, \mathbb{U}] \leftarrow$ FactLU($\mathbb{A}$) ▷ Factorisation $\mathbb{LU}$
3: $\boldsymbol{y} \leftarrow$ RSLTriInf($\mathbb{L}, \boldsymbol{b}$) ▷ Résolution du système $\mathbb{L}\boldsymbol{y} = \boldsymbol{b}$
4: $\boldsymbol{x} \leftarrow$ RSLTriSup($\mathbb{U}, \boldsymbol{y}$) ▷ Résolution du système $\mathbb{U}\boldsymbol{x} = \boldsymbol{y}$
5: **Fin Fonction**

Il nous faut donc écrire la fonction FactLU (les deux fonctions RSLTriInf et RSLTriSup ayant déjà été écrites).

Détermination des matrices $\mathbb{L}$ et $\mathbb{U}$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{R})$ une matrice dont les sous-matrices principales sont inversibles. D'après le théorème 4.5, il existe une unique matrice $\mathbb{L} \in \mathcal{M}_n(\mathbb{R})$ triangulaire inférieure avec $L_{i,i} = 1$, $\forall i \in [\![1, n]\!]$, et une unique matrice $\mathbb{L} \in \mathcal{M}_n(\mathbb{R})$ triangulaire supérieure telles que

$$\mathbb{A} = \mathbb{LU} \tag{4.19}$$ {LU:eq-0}

c'est à dire

$$\begin{pmatrix} A_{1,1} & \dots & A_{1,n} \\ \vdots & \ddots & \dots \\ A_{n,1} & \dots & A_{n,n} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ L_{2,1} & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ L_{n,1} & \dots & L_{n,n-1} & 1 \end{pmatrix} \begin{pmatrix} U_{1,1} & \dots & \dots & U_{n,1} \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & U_{n,n} \end{pmatrix}. \tag{4.20}$$ {LU:eq-1}

Pour déterminer les matrices $\mathbb{L}$ et $\mathbb{U}$, on remarque que la 1ère ligne de $\mathbb{L}$ est déjà déterminée. On peut

alors l'utiliser pour calculer la première ligne de $\mathbb{U}$: $\forall j \in [\![1,n]\!]$

$$\begin{aligned} A_{1,j} &= \sum_{k=1}^{n} L_{1,k}U_{k,j} \\ &= L_{1,1}U_{1,j} \text{ car } \mathbb{L} \text{ triangulaire inférieure} \\ &= U_{1,j} \end{aligned}$$

On a donc

{LU-eqa1}
$$U_{1,j} = A_{1,j}, \quad \forall j \in [\![1,n]\!]. \tag{4.21}$$

La première colonne de $\mathbb{U}$ est aussi déterminée, on peut alors l'utiliser pour calculer la première colonne de $\mathbb{L}$: $\forall i \in [\![2,n]\!]$

$$\begin{aligned} A_{i,1} &= \sum_{k=1}^{n} L_{i,k}U_{1,j} \\ &= L_{i,1}U_{1,1} \text{ car } \mathbb{U} \text{ triangulaire supérieure} \end{aligned}$$

On peut démontrer, de part les hypothèses sur la matrice $\mathbb{A}$, que $U_{1,1} \neq 0$ et alors

{LU-eqa2}
$$L_{1,j} = A_{i,j}/U_{1,1}, \quad \forall i \in [\![2,n]\!]. \tag{4.22}$$

La première ligne de $\mathbb{U}$ et la première colonne de $\mathbb{L}$ sont donc déterminées par les formules (4.21) et (4.22).

Par récurrence, on suppose connues les $i-1$ premières lignes de $\mathbb{U}$ et les $i-1$ premières colonnes de $\mathbb{L}$. On va montrer que l'on peux expliciter la i-ème ligne de $\mathbb{U}$ et la i-ème colonne de $\mathbb{L}$.
En effet, $\forall j \in [\![i,n]\!]$, on a

$$\begin{aligned} A_{i,j} &= \sum_{k=1}^{n} L_{i,k}U_{k,j} \\ &= \sum_{k=1}^{i-1} L_{i,k}U_{k,j} + L_{i,i}U_{i,j} + \sum_{k=i+1}^{n} L_{i,k}U_{k,j} \\ &= \sum_{k=1}^{i-1} L_{i,k}U_{k,j} + L_{i,i}U_{i,j} \text{ car } \mathbb{L} \text{ triangulaire inférieure} \end{aligned}$$

Dans l'expression $\sum_{k=1}^{i-1} L_{i,k}U_{k,j}$ tous les termes sont connus (hypothèse de récurrence) et $L_{i,i} = 1$. On en déduit,

{LU-eqa3}
$$\text{Pour } i \text{ allant de 1 à } n : \quad U_{i,j} = \begin{cases} A_{i,j} - \sum_{k=1}^{i-1} L_{i,k}U_{k,j}, & \forall j \in [\![i,n]\!]. \\ 0, & \forall j \in [\![1,i-1]\!]. \end{cases} \tag{4.23}$$

Maintenant, on calcule, $\forall j \in [\![i+1,n]\!]$,

$$\begin{aligned} A_{j,i} &= \sum_{k=1}^{n} L_{j,k}U_{k,i} \\ &= \sum_{k=1}^{i-1} L_{j,k}U_{k,i} + L_{j,i}U_{i,i} + \sum_{k=i+1}^{n} L_{j,k}U_{k,i} \\ &= \sum_{k=1}^{i-1} L_{j,k}U_{k,i} + L_{j,i}U_{i,i} \text{ car } \mathbb{U} \text{ triangulaire supérieure} \end{aligned}$$

Dans l'expression $\displaystyle\sum_{k=1}^{i-1} L_{j,k}U_{k,i}$ tous les termes sont connus (hypothèse de récurrence). De plus $U_{i,i}$ est donné par (4.23) et on peut démontrer, de part les hypothèses sur la matrice $\mathbb{A}$, que $U_{i,i} \neq 0$. On a alors

{LU-eqa4}
$$\text{Pour } i \text{ allant de 1 à } n : \quad L_{j,i} = \begin{cases} 0, & \forall j \in [\![1,i-1]\!]. \\ 1, & j = i \\ \frac{1}{U_{i,i}} \left(A_{j,i} - \displaystyle\sum_{k=1}^{i-1} L_{j,k}U_{k,i} \right), & \forall j \in [\![i+1,n]\!], \end{cases} \tag{4.24}$$

On écrit en détail les raffinements successifs permettant d'aboutir à l'algorithme final de telle sorte que le passage entre deux raffinements successifs soit le plus compréhensible possible.

Algorithme 4.10 $\mathcal{R}_0$

1: Calculer les matrices et

Algorithme 4.10 $\mathcal{R}_1$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: Calculer la ligne i de .
3: Calculer la colonne i de .
4: **Fin Pour**

Algorithme 4.10 $\mathcal{R}_1$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: Calculer la ligne i de .
3: Calculer la colonne i de .
4: **Fin Pour**

Algorithme 4.10 $\mathcal{R}_2$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
3: $U(i,j) \leftarrow 0$
4: **Fin Pour**
5: **Pour** $j \leftarrow i$ **à** n **faire**
6: $U_{i,j} \leftarrow A_{i,j} - \sum_{k=1}^{i-1} L_{i,k}U_{k,j}$
7: **Fin Pour**
8: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
9: $L_{j,i} \leftarrow 0$
10: **Fin Pour**
11: $L_{i,i} \leftarrow 1$
12: **Pour** $j \leftarrow i+1$ **à** n **faire**
13: $L_{j,i} \leftarrow \frac{1}{U_{i,i}} \left(A_{j,i} - \sum_{k=1}^{i-1} L_{j,k}U_{k,i} \right)$
14: **Fin Pour**
15: **Fin Pour**

Algorithme 4.10 $\mathcal{R}_2$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
3: $U(i,j) \leftarrow 0$
4: **Fin Pour**
5: **Pour** $j \leftarrow i$ **à** n **faire**
6: $U_{i,j} \leftarrow A_{i,j} - \sum_{k=1}^{i-1} L_{i,k}U_{k,j}$
7: **Fin Pour**
8: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
9: $L_{j,i} \leftarrow 0$
10: **Fin Pour**
11: $L_{i,i} \leftarrow 1$
12: **Pour** $j \leftarrow i+1$ **à** n **faire**
13: $L_{j,i} \leftarrow \frac{1}{U_{i,i}} \left(A_{j,i} - \sum_{k=1}^{i-1} L_{j,k}U_{k,i} \right)$
14: **Fin Pour**
15: **Fin Pour**

Algorithme 4.10 $\mathcal{R}_3$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
3: $U(i,j) \leftarrow 0$
4: **Fin Pour**
5: **Pour** $j \leftarrow i$ **à** n **faire**
6: $S_1 \leftarrow \sum_{k=1}^{i-1} L_{i,k}U_{k,j}$ {FC:algo-IB}
7: $U_{i,j} \leftarrow A_{i,j} - S_1$ {FC:algo-IC}
8: **Fin Pour**
9: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
10: $L_{j,i} \leftarrow 0$
11: **Fin Pour**
12: $L_{i,i} \leftarrow 1$
13: **Pour** $j \leftarrow i+1$ **à** n **faire**
14: $S_2 \leftarrow \sum_{k=1}^{i-1} L_{j,k}U_{k,i}$ {FC:algo-ID}
15: $L_{j,i} \leftarrow \frac{1}{U_{i,i}} (A_{j,i} - S_2)$. {FC:algo-IE}
16: **Fin Pour**
17: **Fin Pour**

Algorithme 4.10 $\mathcal{R}_3$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
3: $U(i,j) \leftarrow 0$
4: **Fin Pour**
5: **Pour** $j \leftarrow i$ **à** n **faire**
6: $S_1 \leftarrow \sum_{k=1}^{i-1} L_{i,k}U_{k,j}$
7: $U_{i,j} \leftarrow A_{i,j} - S_1$
8: **Fin Pour**
9: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
10: $L_{j,i} \leftarrow 0$
11: **Fin Pour**
12: $L_{i,i} \leftarrow 1$
13: **Pour** $j \leftarrow i+1$ **à** n **faire**
14: $S_2 \leftarrow \sum_{k=1}^{i-1} L_{j,k}U_{k,i}$
15: $L_{j,i} \leftarrow \frac{1}{U_{i,i}}\left(A_{j,i} - S_2\right).$
16: **Fin Pour**
17: **Fin Pour**

Algorithme 4.10 $\mathcal{R}_4$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
3: $U(i,j) \leftarrow 0$
4: **Fin Pour**
5: **Pour** $j \leftarrow i$ **à** n **faire**
6: $S_1 \leftarrow 0$
7: **Pour** $k \leftarrow 1$ **à** $i-1$ **faire**
8: $S_1 \leftarrow S_1 + L_{i,k} * U_{k,j}$
9: **Fin Pour**
10: $U_{i,j} \leftarrow A_{i,j} - S_1$
11: **Fin Pour**
12: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
13: $L_{j,i} \leftarrow 0$
14: **Fin Pour**
15: $L_{i,i} \leftarrow 1$
16: **Pour** $j \leftarrow i+1$ **à** n **faire**
17: $S_2 \leftarrow 0$
18: **Pour** $k \leftarrow 1$ **à** $i-1$ **faire**
19: $S_2 \leftarrow S_2 + L_{j,k} * U_{k,i}$
20: **Fin Pour**
21: $L_{j,i} \leftarrow \frac{1}{U_{i,i}}\left(A_{j,i} - S_2\right).$
22: **Fin Pour**
23: **Fin Pour**

L'algorithme peut être amélioré, pour gagner en lisibilité... En effet, il est possible d'initialiser la matrice $\mathbb{U}$ par la matrice nulle et la matrice $\mathbb{L}$ par la matrice identitée, ce qui permet alors de supprimer les boucles $U(i,j) \leftarrow 0$ et $L(j,i) \leftarrow 0$ ainsi que la commande $L(i,i) \leftarrow 1$. On obtient alors l'algorithme final

Algorithme 4.10 Fonction FactLU permet de calculer les matrices $\mathbb{L}$ et $\mathbb{U}$ dites matrice de factorisation $\mathbb{LU}$ associée à la matrice $\mathbb{A}$, telle que

$$\mathbb{A} = \mathbb{LU}$$

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{R})$ dont les sous-matrices principales sont inversibles.

Résultat : $\mathbb{L}$: matrice de $\mathcal{M}_n(\mathbb{R})$ triangulaire inférieure avec $L_{i,i} = 1,\ \forall i \in [\![1, n]\!]$

$\mathbb{U}$: matrice de $\mathcal{M}_n(\mathbb{R})$ triangulaire supérieure.

Fonction $[\mathbb{L}, \mathbb{U}] \leftarrow$ FactLU($\mathbb{A}$)
  $\mathbb{U} \leftarrow \mathbb{O}_n$ ▷ $\mathbb{O}_n$ matrice nulle $n \times n$
  $\mathbb{L} \leftarrow \mathbb{I}_n$ ▷ $\mathbb{I}_n$ matrice identitée $n \times n$
  Pour $i \leftarrow 1$ **à** n **faire**
    Pour $j \leftarrow i$ **à** n **faire** ▷ Calcul de la ligne i de $\mathbb{U}$
      $S_1 \leftarrow 0$
      Pour $k \leftarrow 1$ **à** $i-1$ **faire**
        $S_1 \leftarrow S_1 + L(i,k) * U(k,j)$
      Fin Pour
      $U(i,j) \leftarrow A(i,j) - S_1$
    Fin Pour
    Pour $j \leftarrow i+1$ **à** n **faire** ▷ Calcul de la colonne i de $\mathbb{L}$
      $S_2 \leftarrow 0$
      Pour $k \leftarrow 1$ **à** $i-1$ **faire**
        $S_2 \leftarrow S_2 + L(j,k) * U(k,i)$
      Fin Pour
      $L(j,i) \leftarrow (A_{j,i} - S_2)/U(i,i)$.
    Fin Pour
  Fin Pour
Fin Fonction

Remarque 4.9 Pour optimiser en mémoire cette fonction, il possible de stocker les matrices $\mathbb{L}$ et $\mathbb{U}$ dans une même matrice de $\mathcal{M}_n(\mathbb{R})$ ...

4.2.4 Factorisation $\mathbb{LDL}^*$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice hermitienne inversible admettant une factorisation $\mathbb{LU}$. On note $\mathbb{D}$ la matrice diagonale inversible définie par $\mathbb{D} = \operatorname{diag} \mathbb{U}$ (i.e. $D_{i,i} = U_{i,i} \neq 0,\ \forall i \in [\![1, n]\!]$) et $\mathbb{R} = \mathbb{D}^{-1}\mathbb{U}$. La matrice $\mathbb{R}$ est donc matrice triangulaire supérieure à diagonale unité car

$$R_{i,i} = (\mathbb{D}^{-1}\mathbb{U})_{i,i} = \sum_{k=1}^{n} (\mathbb{D}^{-1})_{i,k} U_{k,i} = (\mathbb{D}^{-1})_{i,i} U_{i,i} = \frac{1}{U_{i,i}} U_{i,i} = 1.$$

On a alors

$$\mathbb{A} = \mathbb{LU} = \mathbb{LDD}^{-1}\mathbb{U} = \mathbb{LDR}.$$

De plus comme $\mathbb{A}$ est hermitienne on a

$$\mathbb{A}^* = \mathbb{A} \iff \mathbb{R}^*\mathbb{D}^*\mathbb{L}^* = \mathbb{LDR}$$

Ce qui donne

$$\mathbb{A} = \mathbb{R}^*(\mathbb{D}^*\mathbb{L}^*) = \mathbb{L}(\mathbb{DR})$$

et donc par unicité de la factorisation $\mathbb{LU}$ on a $\mathbb{R}^* = \mathbb{L}$ et $\mathbb{D}^*\mathbb{L}^* = \mathbb{DR}$. On obtient alors

$$\mathbb{R}^* = \mathbb{L} \text{ et } \mathbb{D}^* = \mathbb{D}$$

et on a le théorème suivant

Théorème 4.10: Factorisation $\mathbb{L}\mathbb{D}\mathbb{L}^*$ ★★★★★

{labRSL:MD:th

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice hermitienne inversible admettant une factorisation $\mathbb{L}\mathbb{U}$. Alors $\mathbb{A}$ s'écrit sous la forme

$$\mathbb{A} = \mathbb{L}\mathbb{D}\mathbb{L}^* \tag{4.25}$$

où $\mathbb{D} = \operatorname{diag} \mathbb{U}$ est une matrice à coefficients réels.

Corollaire 4.11: ★★★★★

L:MD:theo:factLDLt}

Une matrice $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ admet une factorisation $\mathbb{L}\mathbb{D}\mathbb{L}^*$ avec $\mathbb{L} \in \mathcal{M}_n(\mathbb{C})$ matrice triangulaire inférieure à diagonale unité et $\mathbb{D} \in \mathcal{M}_n(\mathbb{R})$ matrice diagonale à coeffcients diagonaux strictement positifs **si et seulement si** la matrice $\mathbb{A}$ est hermitienne définie positive.

Proof. $\boxed{\Longrightarrow}$ Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ admettant une factorisation $\mathbb{L}\mathbb{D}\mathbb{L}^*$ avec $\mathbb{L} \in \mathcal{M}_n(\mathbb{C})$ matrice triangulaire inférieure à diagonale unité et $\mathbb{D} \in \mathcal{M}_n(\mathbb{R})$ matrice diagonale à coeffcients diagonaux strictement positifs.
La matrice $\mathbb{A}$ est alors hermitienne car

$$\mathbb{A}^* = (\mathbb{L}\mathbb{D}\mathbb{L}^*)^* = \mathbb{L}^{**}\mathbb{D}^*\mathbb{L}^* = \mathbb{L}\mathbb{D}\mathbb{L}^*.$$

De plus $\forall \boldsymbol{x} \in \mathbb{C}^n \backslash \{0\}$ on a

$$\langle \mathbb{A}\boldsymbol{x}, \boldsymbol{x} \rangle = \langle \mathbb{L}\mathbb{D}\mathbb{L}^*\boldsymbol{x}, \boldsymbol{x} \rangle = \langle \mathbb{D}\mathbb{L}^*\boldsymbol{x}, \mathbb{L}^*\boldsymbol{x} \rangle$$

On pose $\boldsymbol{y} = \mathbb{L}^*\boldsymbol{x} \neq 0$ car $\boldsymbol{x} \neq 0$ et $\mathbb{L}^*$ inversible. On obtient alors

$$\langle \mathbb{A}\boldsymbol{x}, \boldsymbol{x} \rangle = \langle \mathbb{D}\boldsymbol{y}, \boldsymbol{y} \rangle = \sum_{i=1}^{n} D_{i,i}|y_i|^2 > 0$$

car $\mathbb{D}$ diagonale, $D_{i,i} > 0$, $\forall i \in [\![1, n]\!]$ et $\boldsymbol{y} \neq 0$.
La matrice hermitienne $\mathbb{A}$ est donc bien définie positive.

$\boxed{\Longleftarrow}$ Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice hermitienne définie positive.
D'après le Corollaire 4.8, page 84, la matrice $\mathbb{A}$ admet une unique factorisation $\mathbb{L}\mathbb{U}$ et donc d'après le Théorème 4.11, page 90, la matrice hermitienne $\mathbb{A}$ peut s'écrire sous la forme $\mathbb{A} = \mathbb{L}\mathbb{D}\mathbb{L}^*$ où $\mathbb{D}$ est diagonale à coefficients réels et $\mathbb{L}$ triangulaire inférieure à diagonale unité. Il reste à démontrer que $D_{i,i} > 0$, $\forall i \in [\![1, n]\!]$.
Comme $\mathbb{A}$ est définie positive, on a $\forall \boldsymbol{x} \in \mathbb{C}^n \backslash \{0\}$, $\langle \mathbb{A}\boldsymbol{x}, \boldsymbol{x} \rangle > 0$. Or on a

$$\langle \mathbb{A}\boldsymbol{x}, \boldsymbol{x} \rangle = \langle \mathbb{L}\mathbb{D}\mathbb{L}^*\boldsymbol{x}, \boldsymbol{x} \rangle = \langle \mathbb{D}\mathbb{L}^*\boldsymbol{x}, \mathbb{L}^*\boldsymbol{x} \rangle$$

On note $\{\boldsymbol{e}_1, \cdots, \boldsymbol{e}_n\}$, la base canonique de $\mathbb{C}^n$ et on rappelle que $\forall i \in [\![1, n]\!]$, $\langle \mathbb{D}\boldsymbol{e}_i, \boldsymbol{e}_i \rangle = D_{i,i}$. Soit $i \in [\![1, n]\!]$. En choisissant $\boldsymbol{x} = (\mathbb{L}^*)^{-1}\boldsymbol{e}_i \neq 0$, on obtient alors

$$\langle \mathbb{D}\mathbb{L}^*\boldsymbol{x}, \mathbb{L}^*\boldsymbol{x} \rangle = \langle \mathbb{D}\boldsymbol{e}_i, \boldsymbol{e}_i \rangle = D_{i,i} > 0.$$

□

4.2.5 Factorisation de Cholesky

Definition 4.12

MD:Cholesky:def:01}

Une **factorisation régulière de Cholesky** d'une matrice $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ est une factorisation $\mathbb{A} = \mathbb{B}\mathbb{B}^*$ où $\mathbb{B}$ est une matrice triangulaire inférieure inversible.
Si les coefficients diagonaux de $\mathbb{B}$ sont positifs, on parle alors d'une **factorisation positive de Cholesky**.

esky:theo:01}

Théorème 4.13: Factorisation de Cholesky ★★★★★

La matrice $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ admet une factorisation régulière de Cholesky **si et seulement si** la matrice $\mathbb{A}$ est hermitienne définie positive. Dans ce cas, elle admet une unique factorisation positive.

Proof. $\boxed{\Longrightarrow}$ Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ admettant une factorisation régulière de Cholesky $\mathbb{A} = \mathbb{B}\mathbb{B}^*$ avec $\mathbb{B}$ est une matrice triangulaire inférieure inversible.
La matrice $\mathbb{A}$ est hermitienne car

$$\mathbb{A}^* = (\mathbb{B}\mathbb{B}^*)^* = (\mathbb{B}^*)^*\mathbb{B}^* = \mathbb{B}\mathbb{B}^* = \mathbb{A}.$$

Soit $\boldsymbol{x} \in \mathbb{C}^n \backslash \{0\}$, on a

$$\langle \mathbb{A}\boldsymbol{x}, \boldsymbol{x} \rangle = \langle \mathbb{B}\mathbb{B}^*\boldsymbol{x}, \boldsymbol{x} \rangle = \langle \mathbb{B}^*\boldsymbol{x}, \mathbb{B}^*\boldsymbol{x} \rangle = \|\mathbb{B}^*\boldsymbol{x}\|^2 > 0$$

car $\mathbb{B}^*\boldsymbol{x} \neq 0$ ($\mathbb{B}^*$ inversible et $\boldsymbol{x} \neq 0$). Donc la matrice $\mathbb{A}$ est bien hermitienne définie positive.

$\boxed{\Longleftarrow}$ Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice hermitienne définie positive.
D'après le Corollaire 4.11, page 90, il existe alors une matrice $\mathbb{L} \in \mathcal{M}_n(\mathbb{C})$ triangulaire inférieure à diagonale unité et une matrice $\mathbb{D} \in \mathcal{M}_n(\mathbb{R})$ diagonale à coefficient strictement positifs telles que

$$\mathbb{A} = \mathbb{L}\mathbb{D}\mathbb{L}^*.$$

On note $\mathbb{H}\mathcal{M}_n(\mathbb{R})$ une matrice diagonale inversible vérifiant $\mathbb{H}^2 = \mathbb{D}$ (i.e. $H_{i,i} = \pm D_{i,i} \neq 0$, $\forall i \in [\![1, n]\!]$). On a alors

$$\mathbb{A} = \mathbb{L}\mathbb{H}\mathbb{H}\mathbb{L}^* = (\mathbb{L}\mathbb{H})(\mathbb{L}\mathbb{H})^*$$

En posant $\mathbb{B} = \mathbb{L}\mathbb{H}$, la matrice $\mathbb{B}$ est bien triangulaire inférieure inversible car produit d'une matrice triangulaire inférieure inversible par une matrice diagonale inversible et on a $\mathbb{A} = \mathbb{B}\mathbb{B}^*$.

Montrons qu'une factorisation positive de Cholesky est unique.
Soient $\mathbb{B}_1$ et $\mathbb{B}_2$ deux factorisations positives de la matrice $\mathbb{A}$, on a donc

$$\mathbb{A} = \mathbb{B}_1\mathbb{B}_1^* = \mathbb{B}_2\mathbb{B}_2^*.$$

En multipliant à gauche par $\mathbb{B}_2^{-1}$ et à droite par $(\mathbb{B}_1^*)^{-1}$ cette équation on obtient

$$\mathbb{B}_2^{-1}\mathbb{B}_1 = \mathbb{B}_2^*(\mathbb{B}_1^*)^{-1} = \mathbb{B}_2^*(\mathbb{B}_1^{-1})^* = (\mathbb{B}_1^{-1}\mathbb{B}_2)^*$$

En notant $\mathbb{G} = \mathbb{B}_2^{-1}\mathbb{B}_1$, on tire de l'équation précédente

$$\mathbb{G} = (\mathbb{G}^{-1})^*. \tag{4.26}$$ {RSL:MD:Cho

On déduit de la (voir Proposition A.39, page 120), que l'inverse d'une matrice triangulaire inférieure à coefficients diagonaux réels strictement positifs est aussi une matrice triangulaire inférieure à coefficients diagonaux réels strictement positifs. De la (voir Proposition A.38, page 120), on obtient que le produit de matrices triangulaires inférieures à coefficients diagonaux réels strictement positifs reste triangulaire inférieure à coefficients diagonaux réels strictement positifs, on en déduit que les matrices $\mathbb{G} = \mathbb{B}_2^{-1}\mathbb{B}_1$ et $\mathbb{G}^{-1} = \mathbb{B}_1^{-1}\mathbb{B}_2$ sont triangulaires inférieures à coefficients diagonaux réels strictement positifs. Or l'équation (4.26) identifie la matrice triangulaire inférieure $\mathbb{G}$ à la matrice triangulaire supérieure $(\mathbb{G}^{-1})^*$: ce sont donc des matrices diagonales à coefficients diagonaux réels strictement positifs et on a alors $(\mathbb{G}^{-1})^* = \mathbb{G}^{-1}$. De l'équation (4.26), on obtient alors $\mathbb{G} = \mathbb{G}^{-1}$, c'est à dire $\mathbb{G} = \mathbb{I} = \mathbb{B}_2^{-1}\mathbb{B}_1$ et donc $\mathbb{B}_1 = \mathbb{B}_2$. □

Résolution d'un système linéaire par la factorisation de Cholesky

Pour commencer, avec la factorisation de Cholesky point de salut sans matrice hermitienne définie positive!

⚠ N'utiliser la factorisation de Cholesky pour la résolution d'un système linéaire que si la matrice du système est hermitienne définie positive.

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice hermitienne définie positive et $\boldsymbol{b} \in \mathbb{C}^n$. Grâce au théorème 4.13, on obtient :

Trouver $\boldsymbol{x} \in \mathbb{C}^n$ tel que

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}. \tag{4.27}$$

{Ex1:eq-1}

est équivalent à

Trouver $\boldsymbol{x} \in \mathbb{C}^n$ solution de

{Ex1:eq-2}
$$\mathbb{B}^*\boldsymbol{x} = \boldsymbol{y} \tag{4.28}$$

avec $\mathbb{B}$ la matrice de factorisation positive de Cholesky de la matrice $\mathbb{A}$ et $\boldsymbol{y} \in \mathbb{R}^n$ solution de

{Ex1:eq-3}
$$\mathbb{B}\boldsymbol{y} = \boldsymbol{b}. \tag{4.29}$$

On est donc ramené à

Algorithme 4.11 Algorithme de base permettant de résoudre, par une factorisation de Cholesky positive, le système linéaire

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$$

où $\mathbb{A}$ une matrice de $\mathcal{M}_n(\mathbb{C})$ hermitienne définie positive et $\boldsymbol{b} \in \mathbb{C}^n$.

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{R})$ hermitienne définie positive,
$\boldsymbol{b}$: vecteur de $\mathbb{C}^n$.
Résultat : $\boldsymbol{x}$: vecteur de $\mathbb{C}^n$.

1: Trouver la factorisation positive de Cholesky $\mathbb{L}$ de la matrice $\mathbb{A}$,
2: résoudre le système triangulaire inférieur $\mathbb{L}\boldsymbol{y} = \boldsymbol{b}$,
3: résoudre le système triangulaire supérieur $\mathbb{L}^t\boldsymbol{x} = \boldsymbol{y}$.

Ceci permet donc de découper le problème initial en trois sous-problèmes plus simples. De plus, ceux-ci peuvent se traiter de manière indépendante.

Algorithme 4.12 Fonction RSLCholesky permettant de résoudre, par une factorisation de Cholesky positive, le système linéaire

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$$

où $\mathbb{A}$ une matrice symétrique de $\mathcal{M}_n(\mathbb{R})$ définie positive et $\boldsymbol{b} \in \mathbb{R}^n$.

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{R})$ symétrique définie positive,
$\boldsymbol{b}$: vecteur de $\mathbb{R}^n$.
Résultat : $\boldsymbol{x}$: vecteur de $\mathbb{R}^n$.

1: **Fonction** $\boldsymbol{x} \leftarrow$ RSLCholesky($\mathbb{A}, \boldsymbol{b}$)
2: $\mathbb{L} \leftarrow$ Cholesky($\mathbb{A}$) ▷ Factorisation positive de Cholesky
3: $\boldsymbol{y} \leftarrow$ RSLTriInf($\mathbb{L}, \boldsymbol{b}$) ▷ Résolution du système $\mathbb{L}\boldsymbol{y} = \boldsymbol{b}$
4: $\mathbb{U} \leftarrow$ transpose($\mathbb{L}$) ▷ Calcul de la transposé de $\mathbb{L}$
5: $\boldsymbol{x} \leftarrow$ RSLTriSup($\mathbb{U}, \boldsymbol{y}$) ▷ Résolution du système $\mathbb{L}^t\boldsymbol{x} = \boldsymbol{y}$
6: **Fin Fonction**

Il nous faut donc écrire la fonction Cholesky (les deux fonctions RSLTriInf et RSLTriSup ayant déjà été écrites et la fonction transpose étant simple à écrire).

Factorisation positive de Cholesky : écriture de l'algorithme

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice hermitienne définie positive. D'après le Théorème 4.13, il existe une unique matrice $\mathbb{B} \in \mathcal{M}_n(\mathbb{C})$ triangulaire inférieure avec $B_{i,i} \in \mathbb{R}^{+*}$, $\forall i \in [\![1, n]\!]$, telle que

lesky:algo:eq00}
$$\mathbb{A} = \mathbb{B}\mathbb{B}^* \tag{4.30}$$

c'est à dire

$$\begin{pmatrix} A_{1,1} & \dots & A_{1,n} \\ \vdots & \ddots & \dots \\ A_{n,1} & \dots & A_{n,n} \end{pmatrix} = \begin{pmatrix} B_{1,1} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ B_{n,1} & \dots & \dots & B_{n,n} \end{pmatrix} \begin{pmatrix} \overline{B}_{1,1} & \dots & \dots & \overline{B}_{n,1} \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \overline{B}_{n,n} \end{pmatrix}. \tag{4.31}$$

algo:eq01}

Pour déterminer la matrice $\mathbb{B}$, on commence par calculer $B_{1,1}$ (la 1ère ligne de $\mathbb{B}$ est donc déterminée) ce qui nous permet de calculer la 1ère colonne de $\mathbb{B}$.
Ensuite, on calcule $B_{2,2}$ (la 2ème ligne de $\mathbb{B}$ est donc déterminée) ce qui nous permet de calculer la 2ème colonne de $\mathbb{B}$. Etc ...

On écrit en détail les raffinements successifs permettant d'aboutir à l'algorithme final de telle manière que le passage entre deux raffinements successifs soit le plus simple possible.

Algorithme 4.13 $\mathcal{R}_0$

1: Calculer la matrice

Algorithme 4.13 $\mathcal{R}_1$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: Calculer $B_{i,i}$, connaissant les $i-1$ premières colonnes de .
3: Calculer la $i^{\text{ème}}$ colonne de .
4: **Fin Pour**

Pour calculer $B_{i,i}$, connaissant les $i-1$ premières colonnes de $\mathbb{B}$. on utilise (4.30) :

$$A_{i,i} = \sum_{j=1}^{n} B_{i,j}(\mathbb{B}^*)_{j,i} = \sum_{j=1}^{n} |B_{i,j}|^2,$$

et comme $\mathbb{L}$ est triangulaire inférieure on obtient

$$A_{i,i} = \sum_{j=1}^{i} |B_{i,j}|^2 = \sum_{j=1}^{i-1} |B_{i,j}|^2 + |B_{i,i}|^2.$$

On a donc

$$B_{i,i} = \left(A_{i,i} - \sum_{j=1}^{i-1} |B_{i,j}|^2 \right)^{1/2}, \ , \ \forall i \in [\![1, n]\!], \tag{4.32}$$

{RSL:MD:Cholesky

Comme les $i-1$ premières colonnes de $\mathbb{B}$ ont déjà été calculées, $B_{i,i}$ est parfaitement déterminée par la formule précédente.

Remarque 4.14 Les hypothèses sur la matrice $\mathbb{A}$ permettent d'affirmer que, $\forall i \in [\![1, n]\!]$, $A_{i,i} - \sum_{j=1}^{i-1} |B_{i,j}|^2 > 0$ et donc $B_{i,i} > 0$.

Pour calculer la $i^{\text{ème}}$ colonne de $\mathbb{B}$, il suffit de déterminer $B_{j,i}$, $\forall j \in [\![i+1, n]\!]$. Pour celà on utilise (4.30)

$$A_{j,i} = \sum_{k=1}^{n} B_{j,k}(\mathbb{B}^*)_{k,i} = \sum_{k=1}^{n} B_{j,k}B_{i,k}, \ \forall j \in [\![i+1, n]\!]$$

Comme $\mathbb{L}$ est triangulaire inférieure on obtient

$$A_{j,i} = \sum_{k=1}^{i} B_{j,k}B_{i,k} = \sum_{k=1}^{i-1} B_{j,k}B_{i,k} + B_{j,i}B_{i,i}, \ \forall j \in [\![i+1, n]\!]$$

Or $B_{i,i} > 0$ vient d'être calculé et les $i-1$ premières colonnes de $\mathbb{L}$ ont déjà été calculées, ce qui donne

$$B_{j,i} = \frac{1}{B_{i,i}} \left(A_{j,i} - \sum_{k=1}^{i-1} B_{j,k}B_{i,k} \right), \ \forall j \in [\![i+1, n]\!] \tag{4.33}$$

{RSL:MD:Cholesky

$$B_{j,i} = 0, \ \forall j \in [\![1, i-1]\!]. \tag{4.34}$$

{RSL:MD:Cholesky

Avec (4.32) et (4.33), on a

Algorithme 4.13 $\mathcal{R}_1$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: Calculer $B_{i,i}$, connaissant les $i-1$ premières colonnes de .
3: Calculer la $i^{\text{ème}}$ colonne de .
4: **Fin Pour**

Algorithme 4.13 $\mathcal{R}_2$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $B_{i,i} \leftarrow \left(A_{i,i} - \sum_{j=1}^{i-1} |B_{i,j}|^2 \right)^{1/2}$
3: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
4: $B_{j,i} \leftarrow 0$
5: **Fin Pour**
6: **Pour** $j \leftarrow i+1$ **à** n **faire**
7: $B_{j,i} \leftarrow \frac{1}{B_{i,i}} \left(A_{j,i} - \sum_{k=1}^{i-1} B_{j,k} B_{i,k} \right)$.
8: **Fin Pour**
9: **Fin Pour**

{FC:algo-GB}

Algorithme 4.13 $\mathcal{R}_2$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $B_{i,i} \leftarrow \left(A_{i,i} - \sum_{j=1}^{i-1} |B_{i,j}|^2 \right)^{1/2}$
3: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
4: $B_{j,i} \leftarrow 0$
5: **Fin Pour**
6: **Pour** $j \leftarrow i+1$ **à** n **faire**
7: $B_{j,i} \leftarrow \frac{1}{B_{i,i}} \left(A_{j,i} - \sum_{k=1}^{i-1} B_{j,k} B_{i,k} \right)$.
8: **Fin Pour**
9: **Fin Pour**

{FC:algo-GA}
{FC:algo-GB}

Algorithme 4.13 $\mathcal{R}_3$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $S_1 \leftarrow \sum_{j=1}^{i-1} |B_{i,j}|^2$
3: $B_{i,i} \leftarrow (A_{i,i} - S_1)^{1/2}$
4: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
5: $B_{j,i} \leftarrow 0$
6: **Fin Pour**
7: **Pour** $j \leftarrow i+1$ **à** n **faire**
8: $S_2 \leftarrow \sum_{k=1}^{i-1} B_{j,k} B_{i,k}$
9: $B_{j,i} \leftarrow \frac{1}{B_{i,i}} (A_{j,i} - S_2)$.
10: **Fin Pour**
11: **Fin Pour**

{FC:algo-IB}
{FC:algo-IC}
{FC:algo-ID}
{FC:algo-IE}

Algorithme 4.13 $\mathcal{R}_3$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $\quad S_1 \leftarrow \sum_{j=1}^{i-1} |B_{i,j}|^2$
3: $\quad B_{i,i} \leftarrow (A_{i,i} - S_1)^{1/2}$
4: $\quad$ **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
5: $\quad\quad B_{j,i} \leftarrow 0$
6: $\quad$ **Fin Pour**
7: $\quad$ **Pour** $j \leftarrow i+1$ **à** n **faire**
8: $\quad\quad S_2 \leftarrow \sum_{k=1}^{i-1} B_{j,k} B_{i,k}$
9: $\quad\quad B_{j,i} \leftarrow \frac{1}{B_{i,i}} (A_{j,i} - S_2)$.
10: $\quad$ **Fin Pour**
11: **Fin Pour**

Algorithme 4.13 $\mathcal{R}_4$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $\quad S_1 \leftarrow 0$
3: $\quad$ **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
4: $\quad\quad S_1 \leftarrow S_1 + |B_{i,j}|^2$
5: $\quad$ **Fin Pour**
6: $\quad B_{i,i} \leftarrow (A_{i,i} - S_1)^{1/2}$
7: $\quad$ **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
8: $\quad\quad B_{j,i} \leftarrow 0$
9: $\quad$ **Fin Pour**
10: $\quad$ **Pour** $j \leftarrow i+1$ **à** n **faire**
11: $\quad\quad S_2 \leftarrow 0$
12: $\quad\quad$ **Pour** $k \leftarrow 1$ **à** $i-1$ **faire**
13: $\quad\quad\quad S_2 \leftarrow S_2 + B_{j,k} B_{i,k}$
14: $\quad\quad$ **Fin Pour**
15: $\quad\quad B_{j,i} \leftarrow \frac{1}{B_{i,i}} (A_{j,i} - S_2)$.
16: $\quad$ **Fin Pour**
17: **Fin Pour**

On obtient alors l'algorithme final

Algorithme 4.13 Fonction CHOLESKY permettant de calculer la matrice $\mathbb{B}$, dites matrice de factorisation positive de Cholesky associée à la matrice $\mathbb{A}$, telle que

$$\mathbb{A} = \mathbb{B}\mathbb{B}^*.$$

Données : `A` : matrice de $\mathcal{M}_n(\mathbb{C})$ hermitienne définie positive.
Résultat : `B` : matrice de $\mathcal{M}_n(\mathbb{C})$ triangulaire inférieure avec $\mathtt{B}(i,i) > 0,\ \forall i \in [\![1,n]\!]$

1: **Fonction** `B` $\leftarrow$ CHOLESKY(`A`)
2: $\quad$ **Pour** $i \leftarrow 1$ **à** n **faire**
3: $\quad\quad S_1 \leftarrow 0$
4: $\quad\quad$ **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
5: $\quad\quad\quad S_1 \leftarrow S_1 + |\mathtt{B}(i,j)|^2$
6: $\quad\quad$ **Fin Pour**
7: $\quad\quad \mathtt{B}(i,i) \leftarrow$ SQRT$(\mathtt{A}(i,i) - S_1)$
8: $\quad\quad$ **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
9: $\quad\quad\quad \mathtt{B}(j,i) \leftarrow 0$
10: $\quad\quad$ **Fin Pour**
11: $\quad\quad$ **Pour** $j \leftarrow i+1$ **à** n **faire**
12: $\quad\quad\quad S_2 \leftarrow 0$
13: $\quad\quad\quad$ **Pour** $k \leftarrow 1$ **à** $i-1$ **faire**
14: $\quad\quad\quad\quad S_2 \leftarrow S_2 + \mathtt{B}(j,k) * \mathtt{B}(i,k)$
15: $\quad\quad\quad$ **Fin Pour**
16: $\quad\quad\quad \mathtt{B}(j,i) \leftarrow (\mathtt{A}(j,i) - S_2)/\mathtt{B}(i,i)$.
17: $\quad\quad$ **Fin Pour**
18: $\quad$ **Fin Pour**
19: **Fin Fonction**

4.2.6 Factorisation $\mathbb{QR}$

La tranformation de Householder

Definition 4.15: Matrice élémentaire de Householder {RSL:MD:QR:de

Soit $\boldsymbol{u} \in \mathbb{C}^n$ tel que $\|\boldsymbol{u}\|_2 = 1$. On appelle **matrice élémentaire de Householder** la matrice $\mathbb{H}(\boldsymbol{u}) \in \mathcal{M}_n(\mathbb{C})$ définie par

$$\mathbb{H}(\boldsymbol{u}) = \mathbb{I} - 2\boldsymbol{u}\boldsymbol{u}^*. \tag{4.35}$$ {RSL:MD:QR:de

Propriété 4.16 {RSL:MD:QR:prop:01}

Toute matrice élémentaire de Householder est hermitienne et unitaire.

Proof. Pour simplifier, on note $\mathbb{H} = \mathbb{H}(\boldsymbol{u})$. Cette matrice est hermitienne car

$$\mathbb{H}^* = (\mathbb{I} - 2\boldsymbol{u}\boldsymbol{u}^*)^* = \mathbb{I} - 2(\boldsymbol{u}\boldsymbol{u}^*)^* = \mathbb{I} - 2\boldsymbol{u}\boldsymbol{u}^* = \mathbb{H}.$$

Montrons qu'elle est unitaire (i.e. $\mathbb{H}^*\mathbb{H} = \mathbb{I}$). On a

$$\begin{aligned}\mathbb{H}^*\mathbb{H} = \mathbb{H}\mathbb{H} &= (\mathbb{I} - 2\boldsymbol{u}\boldsymbol{u}^*)(\mathbb{I} - 2\boldsymbol{u}\boldsymbol{u}^*) \\ &= \mathbb{I} - 4\boldsymbol{u}\boldsymbol{u}^* + 4\boldsymbol{u}\boldsymbol{u}^*\boldsymbol{u}\boldsymbol{u}^*.\end{aligned}$$

Or on a $\boldsymbol{u}^*\boldsymbol{u} = \|\boldsymbol{u}\|_2 = 1$ par hypothèse et donc

$$\mathbb{H}^*\mathbb{H} = \mathbb{I} - 4\boldsymbol{u}\boldsymbol{u}^* + 4\boldsymbol{u}(\boldsymbol{u}^*\boldsymbol{u})\boldsymbol{u}^* = \mathbb{I}.$$

□

Propriété 4.17 {RSL:MD:QR:prop:03}

Soient $\boldsymbol{x} \in \mathbb{K}^n$ et $\boldsymbol{u} \in \mathbb{K}^n$, $\|\boldsymbol{u}\|_2 = 1$. On note $\boldsymbol{x}_{\|} = \mathrm{proj}_{\boldsymbol{u}}(\boldsymbol{x}) \overset{\text{def}}{=} \langle \boldsymbol{u}, \boldsymbol{x}\rangle \boldsymbol{u}$ et $\boldsymbol{x}_{\perp} = \boldsymbol{x} - \boldsymbol{x}_{\|}$. On a alors

$$\mathbb{H}(\boldsymbol{u})(\boldsymbol{x}_{\perp} + \boldsymbol{x}_{\|}) = \boldsymbol{x}_{\perp} - \boldsymbol{x}_{\|}. \tag{4.36}$$

et

$$\mathbb{H}(\boldsymbol{u})\boldsymbol{x} = \boldsymbol{x}, \quad \text{si } \langle \boldsymbol{x}, \boldsymbol{u}\rangle = 0. \tag{4.37}$$

Proof. On note que par construction $\langle \boldsymbol{u}, \boldsymbol{x}_{\perp}\rangle = 0$. On a

$$\begin{aligned}\mathbb{H}(\boldsymbol{u})(\boldsymbol{x}_{\perp} + \boldsymbol{x}_{\|}) = (\mathbb{I} - 2\boldsymbol{u}\boldsymbol{u}^*)(\boldsymbol{x}_{\perp} + \boldsymbol{x}_{\|}) &= \boldsymbol{x}_{\perp} + \boldsymbol{x}_{\|} - 2\boldsymbol{u}\underbrace{\boldsymbol{u}^*\boldsymbol{x}_{\perp}}_{=0} - 2\boldsymbol{u}\boldsymbol{u}^*\boldsymbol{x}_{\|} \\ &= \boldsymbol{x}_{\perp} + \boldsymbol{x}_{\|} - 2\boldsymbol{u}\boldsymbol{u}^*\boldsymbol{u}\langle \boldsymbol{u}, \boldsymbol{x}\rangle = \boldsymbol{x}_{\perp} + \boldsymbol{x}_{\|} - 2\boldsymbol{u}\underbrace{\boldsymbol{u}^*\boldsymbol{u}}_{=1}\boldsymbol{u}^*\boldsymbol{x} \\ &= \boldsymbol{x}_{\perp} + \boldsymbol{x}_{\|} - 2\boldsymbol{u}\boldsymbol{u}^*\boldsymbol{x} = \boldsymbol{x}_{\perp} + \boldsymbol{x}_{\|} - 2\boldsymbol{x}_{\|} \\ &= \boldsymbol{x}_{\perp} - \boldsymbol{x}_{\|}.\end{aligned}$$

Si $\langle \boldsymbol{x}, \boldsymbol{u}\rangle = 0$ alors $\boldsymbol{x}_{\|} = 0$ et $\boldsymbol{x} = \boldsymbol{x}_{\perp}$. □

Théorème 4.18 {RSL:MD:QR:theo:01}

Soient $\boldsymbol{a}$, $\boldsymbol{b}$ deux vecteurs non colinéaires de $\mathbb{C}^n$ avec $\|\boldsymbol{b}\|_2 = 1$. Soit $\alpha \in \mathbb{C}$ tel que $|\alpha| = \|\boldsymbol{a}\|_2$ et $\arg\alpha = -\arg\langle \boldsymbol{a}, \boldsymbol{b}\rangle \ [\pi]$. On a alors

$$\mathbb{H}\left(\frac{\boldsymbol{a} - \alpha\boldsymbol{b}}{\|\boldsymbol{a} - \alpha\boldsymbol{b}\|_2}\right)\boldsymbol{a} = \alpha\boldsymbol{b}. \tag{4.38}$$

Proof. (voir Exercice 4.2.4, page 97) □

Exercice 4.2.4

Soient $\boldsymbol{a}$ et $\boldsymbol{b}$ deux vecteurs non colinéaires de $\mathbb{C}^n$ avec $\|\boldsymbol{b}\|_2 = 1$. On va chercher $\alpha \in \mathbb{C}$ et $\boldsymbol{u} \in \mathbb{C}^n$ vérifiant

$$\mathbb{H}(\boldsymbol{u})\boldsymbol{a} = \alpha\boldsymbol{b}. \tag{4.39}$$

Q. 1 1. *Montrer que si* α *vérifie* (4.39) *alors* $|\alpha| = \|\boldsymbol{a}\|_2$.

2. *Montrer que si* $\arg\alpha = -\arg(\langle \boldsymbol{a}, \boldsymbol{b}\rangle)\ [\pi]$ *alors* $\alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle \in \mathbb{R}$.

Q. 2 *Soient* α *et* $\boldsymbol{u}$ *vérifiant* (4.39).

1. *Montrer que*

$$|\langle \boldsymbol{u}, \boldsymbol{a}\rangle|^2 = \frac{\langle \boldsymbol{a}, \boldsymbol{a}\rangle - \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle}{2} \tag{4.40}$$

2. *Montrer que si* $\arg\alpha = -\arg(\langle \boldsymbol{a}, \boldsymbol{b}\rangle)\ [\pi]$ *alors* $\langle \boldsymbol{a}, \boldsymbol{a}\rangle - \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle \in \mathbb{R}^{*+}$.

3. *En déduire que*

$$\boldsymbol{u} = \frac{1}{2\lambda}(\boldsymbol{a} - \alpha\boldsymbol{b}), \quad \text{avec } \lambda = \pm\left(\frac{\langle \boldsymbol{a}, \boldsymbol{a}\rangle - \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle}{2}\right)^{1/2} \tag{4.41}$$

Correction Exercice 4.2.4 On pose $\mathbb{H} = \mathbb{H}(\boldsymbol{u})$ pour alléger les notations.

Q. 1 1. On a

$$\begin{aligned}\|\boldsymbol{a}\|_2^2 = \langle \boldsymbol{a}, \boldsymbol{a}\rangle &= \langle \mathbb{H}^*\mathbb{H}\boldsymbol{a}, \boldsymbol{a}\rangle \quad \text{car } \mathbb{H} \text{ unitaire}\\ &= \langle \mathbb{H}\boldsymbol{a}, \mathbb{H}\boldsymbol{a}\rangle \quad \text{par définition du produit scalaire}\\ &= \|\mathbb{H}\boldsymbol{a}\|_2^2 = \|\alpha\boldsymbol{b}\|_2^2 = |\alpha|^2\|\boldsymbol{b}\|_2^2 = |\alpha|^2.\end{aligned}$$

2. On a par définition de l'argument $\alpha = |\alpha|e^{\imath\arg\alpha}$ et $\langle \boldsymbol{a}, \boldsymbol{b}\rangle = |\langle \boldsymbol{a}, \boldsymbol{b}\rangle|e^{\imath\arg(\langle \boldsymbol{a}, \boldsymbol{b}\rangle)}$ ce qui donne

$$\alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle = |\alpha||\langle \boldsymbol{a}, \boldsymbol{b}\rangle|e^{\imath(\arg\alpha + \arg(\langle \boldsymbol{a}, \boldsymbol{b}\rangle))} \tag{4.42}$$

et donc $\alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle$ est réel si $\arg\alpha + \arg(\langle \boldsymbol{a}, \boldsymbol{b}\rangle) = 0\ [\pi]$.

Q. 2 1. On a

$$\begin{aligned}\mathbb{H}(\boldsymbol{u})\boldsymbol{a} = \alpha\boldsymbol{b} &\iff (\mathbb{I} - 2\boldsymbol{u}\boldsymbol{u}^*)\boldsymbol{a} = \alpha\boldsymbol{b}\\ &\iff \boldsymbol{a} - 2\boldsymbol{u}(\boldsymbol{u}^*\boldsymbol{a}) = \alpha\boldsymbol{b}\end{aligned}$$

et donc

$$\boldsymbol{a} - 2\langle \boldsymbol{u}, \boldsymbol{a}\rangle\boldsymbol{u} = \alpha\boldsymbol{b} \tag{4.43}$$

En effectuant le produit scalaire avec $\boldsymbol{a}$ de cette dernière équation, on obtient

$$\langle \boldsymbol{a}, \boldsymbol{a}\rangle - 2\langle \boldsymbol{u}, \boldsymbol{a}\rangle\langle \boldsymbol{a}, \boldsymbol{u}\rangle = \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle$$

ce qui prouve (4.40).

2. On a montré en Q.1 que $\alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle \in \mathbb{R}$ et donc $\langle \boldsymbol{a}, \boldsymbol{a}\rangle - \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle \in \mathbb{R}$. Il reste donc à montrer que $\langle \boldsymbol{a}, \boldsymbol{a}\rangle - \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle > 0$.

- Si $\arg\alpha = -\arg(\langle \boldsymbol{a}, \boldsymbol{b}\rangle) + \pi\ [2\pi]$, alors de (4.42) on obtient $\alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle \leqslant 0$ et donc $\langle \boldsymbol{a}, \boldsymbol{a}\rangle - \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle \leqslant \|\boldsymbol{a}\|_2 > 0$ car $\boldsymbol{a} \neq 0$.
- Si $\arg\alpha = -\arg(\langle \boldsymbol{a}, \boldsymbol{b}\rangle)\ [2\pi]$, alors de (4.42) on obtient $\alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle \geqslant 0$.
  Comme les vecteurs $\boldsymbol{a}$ et $\boldsymbol{b}$ ne sont pas colinéaires on a inégalité stricte dans Cauchy-Schwarz :

$$|\langle \boldsymbol{a}, \boldsymbol{b}\rangle| < \|\boldsymbol{a}\|_2\|\boldsymbol{b}\|_2 = \|\boldsymbol{a}\|_2.$$

On obtient donc

$$0 \leqslant \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle \leqslant |\alpha||\langle \boldsymbol{a}, \boldsymbol{b}\rangle| < |\alpha|\|\boldsymbol{a}\|_2 = \|\boldsymbol{a}\|_2^2$$

Attention, dans ce cas $\langle \boldsymbol{a}, \boldsymbol{a}\rangle - \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle$ peut-être très petit.

3. De (4.43), on en déduit immédiatement (4.41).
Vérifions que $\|\boldsymbol{u}\|_2 = 1$. On a

$$\|\boldsymbol{u}\|_2^2 = \langle \boldsymbol{u}, \boldsymbol{u}\rangle = \frac{1}{4|\lambda|^2}\langle \boldsymbol{a} - \alpha\boldsymbol{b}, \boldsymbol{a} - \alpha\boldsymbol{b}\rangle$$

Or

$$\begin{aligned}\langle \boldsymbol{a} - \alpha\boldsymbol{b}, \boldsymbol{a} - \alpha\boldsymbol{b}\rangle &= \langle \boldsymbol{a}, \boldsymbol{a}\rangle - \overline{\alpha}\langle \boldsymbol{b}, \boldsymbol{a}\rangle - \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle + |\alpha|^2\langle \boldsymbol{b}, \boldsymbol{b}\rangle = \|\boldsymbol{a}\|_2^2 - \overline{\alpha}\langle \boldsymbol{b}, \boldsymbol{a}\rangle - \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle + |\alpha|^2 \\ &= 2\|\boldsymbol{a}\|_2^2 - \overline{\alpha}\langle \boldsymbol{b}, \boldsymbol{a}\rangle - \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle \\ &= 2\|\boldsymbol{a}\|_2^2 - 2\alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle \quad \text{car } \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle = \overline{(\alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle)} = \overline{\alpha}\langle \boldsymbol{b}, \boldsymbol{a}\rangle \in \mathbb{R}\end{aligned}$$

De plus

$$\begin{aligned}4|\lambda|^2 &= 2(\langle \boldsymbol{a}, \boldsymbol{a}\rangle - \alpha\langle \boldsymbol{b}, \boldsymbol{a}\rangle) \in \mathbb{R} \\ &= 2\|\boldsymbol{a}\|_2^2 - 2\alpha\langle \boldsymbol{b}, \boldsymbol{a}\rangle\end{aligned}$$

◇

Exercice 4.2.5

{RSL:MD:QR:exo:14}

Soient $\boldsymbol{a}$ et $\boldsymbol{b}$ deux vecteurs non nuls et non colinéaires de $\mathbb{C}^n$ avec $\|\boldsymbol{b}\|_2 = 1$.

Q. 1 *Ecrire la fonction algorithmique* HOUSEHOLDER *permettant de retourner une matrice de Householder* $\mathbb{H}$ *et* $\alpha \in \mathbb{C}$ *tels que* $\mathbb{H}(\boldsymbol{u})\boldsymbol{a} = \alpha\boldsymbol{b}$. *Le choix du* α *est fait par le paramètre* δ *(0 ou 1) de telle sorte que* $\alpha = -\arg(\langle \boldsymbol{a}, \boldsymbol{b}\rangle) + \delta\pi$.
Des fonctions comme DOT$(\boldsymbol{a}, \boldsymbol{b})$ *(produit scalaire de deux vecteurs),* NORM$(\boldsymbol{a})$ *(norme d'un vecteur),* ARG(z) *(argument d'un nombre complexe),* MATPROD$(\mathbb{A}, \mathbb{B})$ *(produit de deux matrices),* CTRANSPOSE$(\mathbb{A})$ *(adjoint d'une matrice), ... pourront être utilisées*

Q. 2 *Proposer un programme permettant de tester cette fonction. On pourra utiliser la fonction* VECRAND(n) *retournant un vecteur aléatoire de* $\mathbb{C}^n$, *les parties réelles et imaginaires de chacune de ses composantes étant dans* $]0, 1[$ *(loi uniforme).*

Q. 3 *Proposer un programme permettant de vérifier que* $\delta = 1$ *est le "meilleur" choix.*

Correction Exercice 4.2.5 Soient $\boldsymbol{a}$ et $\boldsymbol{b}$ deux vecteurs non nuls et non colinéaires de $\mathbb{C}^n$.

Q. 1 Les données du problème sont $\boldsymbol{a}$, $\boldsymbol{b}$ et δ. On veut calculer α et la matrice $\mathbb{H}(\boldsymbol{u})$.

Algorithme 4.14 Calcul du α et de la matrice de Householder $\mathbb{H}(\boldsymbol{u})$ telle que $\mathbb{H}(\boldsymbol{u})\boldsymbol{a} = \alpha\boldsymbol{b}$.

Données : $\boldsymbol{a}, \boldsymbol{b}$: deux vecteurs de $\mathbb{C}^n$ non nuls et non colinéaires.
δ : 0 ou 1, permet de déterminer α.
Résultat : $\mathbb{H}$: matrice de Householder dans $\mathcal{M}_n(\mathbb{C})$,
α : nombre complexe, donné par $-\arg(\langle \boldsymbol{a}, \boldsymbol{b}\rangle) + \delta\pi$.

1: **Fonction** $[\mathbb{H}, \alpha] \leftarrow$ HOUSEHOLDER$(\boldsymbol{a}, \boldsymbol{b}, \delta)$
2: ab $\leftarrow$ DOT(A,B) ▷ DOT produit scalaire dans $\mathbb{C}$.
3: $\alpha \leftarrow$ NORM$(\boldsymbol{a}, 2) * exp(i * (\delta * \pi -$ ARG$(ab)))$
4: $\boldsymbol{u} \leftarrow \boldsymbol{a} - \alpha * \boldsymbol{b}$
5: $\boldsymbol{u} \leftarrow \boldsymbol{u}/$NORM$(\boldsymbol{u}, 2)$
6: $\mathbb{H} \leftarrow$ EYE$(n) - 2 *$ MATPROD$(\boldsymbol{u},$ CTRANSPOSE$(\boldsymbol{u}))$
7: **Fin Fonction**

Q. 2
1: $n \leftarrow 100$
2: $\boldsymbol{a} \leftarrow$ VECRAND(n)
3: $\boldsymbol{b} \leftarrow$ VECRAND(n)
4: $\boldsymbol{b} \leftarrow$ NORM$(\boldsymbol{b}, 2)$

5: $[\mathbb{H}, \alpha] \leftarrow$ HOUSEHOLDER$(\boldsymbol{a}, \boldsymbol{b}, 0)$
6: error $\leftarrow$ NORM$(\mathbb{H} * \boldsymbol{a} - \alpha * \boldsymbol{b}, 2)$

Q. 3
1: $n \leftarrow 100$
2: $\boldsymbol{a} \leftarrow$ VECRAND(n)
3: $\boldsymbol{b} \leftarrow \boldsymbol{a} + 1e-6 *$ VECRAND(n)
4: $\boldsymbol{b} \leftarrow \boldsymbol{b}/$NORM$(\boldsymbol{b}, 2)$
5: $[\mathbb{H}_1, \alpha_1] \leftarrow$ HOUSEHOLDER$(\boldsymbol{a}, \boldsymbol{b}, 1)$
6: $[\mathbb{H}_0, \alpha_0] \leftarrow$ HOUSEHOLDER$(\boldsymbol{a}, \boldsymbol{b}, 0)$
7: error0 $\leftarrow$ NORM$(\mathbb{H}_0 * \boldsymbol{a} - \alpha_0 * \boldsymbol{b}, 2)/ABS(\alpha_0)$
8: error1 $\leftarrow$ NORM$(\mathbb{H}_1 * \boldsymbol{a} - \alpha_1 * \boldsymbol{b}, 2)/ABS(\alpha_1)$

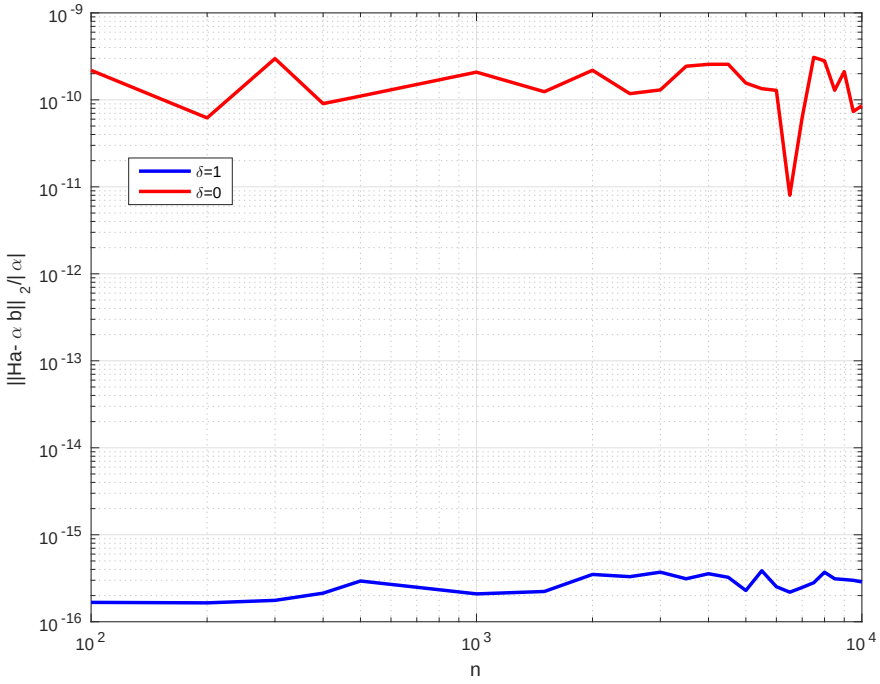


Figure 4.1: Choix de α dans HOUSEHOLDER : erreur relative en norme L_2 RSL:MD:Householder:fig:01

◇

Si l'on souhaite calculer le produit matrice-vecteur $\mathbb{H}(\boldsymbol{u})\boldsymbol{x}$, il n'est pas nécessaire de calculer explicitement la matrice $\mathbb{H}(\boldsymbol{u})$. En effet, on pose $\boldsymbol{v} = 2\lambda\boldsymbol{u} = \boldsymbol{a} - \alpha\boldsymbol{b}$ et $\beta = 2\lambda^2 = \|a\|_2^2 - \alpha\langle \boldsymbol{a}, \boldsymbol{b}\rangle > 0$. On choisi α de manière à maximiser β pour éviter une division par un nombre trop petit. On prend donc

$$\alpha = \|a\|_2 \, e^{\imath(\pi - \arg(\langle \boldsymbol{a}, \boldsymbol{b}\rangle))}$$

On obtient alors

$$\mathbb{H}(\boldsymbol{u})\boldsymbol{x} = \boldsymbol{x} - \frac{1}{\beta}\boldsymbol{v}\boldsymbol{v}^*\boldsymbol{x} = \boldsymbol{x} - \frac{\langle \boldsymbol{v}, \boldsymbol{x}\rangle}{\beta}\boldsymbol{v}. \tag{4.44}$$

Corollaire 4.19

{RSL:MD:QR:co

Soit $\boldsymbol{a} \in \mathbb{C}^n$ avec $a_1 \neq 0$ et $\exists j \in [\![2, n]\!]$ tel que $a_j \neq 0$. Soient $\theta = \arg a_1$ et

$$\boldsymbol{u}_\pm = \frac{\boldsymbol{a} \pm \|\boldsymbol{a}\|_2 e^{\imath\theta} \boldsymbol{e}_1}{\|\boldsymbol{a} \pm \|\boldsymbol{a}\|_2 e^{\imath\theta} \boldsymbol{e}_1\|}$$

Alors

$$\mathbb{H}(\boldsymbol{u}_\pm)\boldsymbol{a} = \mp \|\boldsymbol{a}\|_2 e^{\imath\theta} \boldsymbol{e}_1 \tag{4.45}$$

où $\boldsymbol{e}_1$ désigne le premier vecteur de la base canonique de $\mathbb{C}^n$.

Proof. On va utiliser le Théorème 4.18.

On pose $\alpha = \pm \|\boldsymbol{a}\|_2 e^{\imath\theta}$ et $\boldsymbol{b} = \boldsymbol{e}_1$. Comme $\arg(\langle \boldsymbol{a}, \boldsymbol{b} \rangle) = \arg \overline{a_1} = -\arg a_1 = -\theta$, on a $\arg \alpha = \theta \ [\pi] = -\arg(\langle \boldsymbol{a}, \boldsymbol{b} \rangle) \ [\pi]$.

Les autres hypothèses du Théorème 4.18 sont vérifiées puisque le vecteur $\boldsymbol{a}$ n'est pas colinéaire à $\boldsymbol{e}_1$ et que $\|\boldsymbol{e}_1\|_2 = 1$. On a donc

$$\mathbb{H}\left(\frac{\boldsymbol{a} \mp \|\boldsymbol{a}\|_2 e^{\imath\theta} \boldsymbol{e}_1}{\|\boldsymbol{a} \mp \|\boldsymbol{a}\|_2 e^{\imath\theta} \boldsymbol{e}_1\|}\right) \boldsymbol{a} = \pm \|\boldsymbol{a}\|_2 e^{\imath\theta} \boldsymbol{e}_1.$$

□

Sur le même principe que l'écriture algébrique de la méthode de Gauss, nous allons transformer la matrice $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ en une matrice triangulaire supérieure à l'aide de matrices de Householder. On a le théorème

{RSL:MD:QR:theo:02}

Théorème 4.20

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice. Il existe une matrice unitaire $\mathbb{Q} \in \mathcal{M}_n(\mathbb{C})$ produit d'au plus $n-1$ matrices de Householder et une matrice triangulaire supérieure $\mathbb{R} \in \mathcal{M}_n(\mathbb{C})$ telles que

$$\mathbb{A} = \mathbb{QR}. \tag{4.46}$$

Si $\mathbb{A}$ est réelle alors $\mathbb{Q}$ et $\mathbb{R}$ sont aussi réelles et l'on peut choisir $\mathbb{Q}$ de telle sorte que les coefficients diagonaux de $\mathbb{R}$ soient positifs. De plus, si $\mathbb{A}$ est inversible alors la factorisation est unique.

Exercice 4.2.6

{ID:QR:exo:15}

Soit $\mathbb{B} \in \mathcal{M}_{m+n}(\mathbb{K})$ la matrice bloc

$$\mathbb{B} = \left(\begin{array}{c|c} \mathbb{B}_{1,1} & \mathbb{B}_{1,2} \\ \hline \mathbb{B}_{2,1} & \mathbb{S} \end{array}\right)$$

où $\mathbb{B}_{1,1} \in \mathcal{M}_m(\mathbb{K})$ et $\mathbb{S} \in \mathcal{M}_n(\mathbb{K})$. On note $\boldsymbol{s} \in \mathbb{K}^n$ le premier vecteur colonne de $\mathbb{S}$ et on suppose que $\boldsymbol{s} \neq 0$ et $\boldsymbol{s}$ non colinéaire à $\boldsymbol{e}_1^n$ premier vecteur de la base canonique de $\mathbb{K}^n$.

Q. 1 *1. Montrer qu'il existe une matrice de Householder $\underline{\mathbb{H}} = \mathbb{H}(\underline{\boldsymbol{u}}) \in \mathcal{M}_n(\mathbb{K})$ et $\alpha \in \mathbb{K}^*$ tel que*

$$\underline{\mathbb{H}}\mathbb{S} = \left(\begin{array}{c|ccc} \pm\alpha & \bullet & \cdots & \bullet \\ \hline 0 & \bullet & \cdots & \bullet \\ \vdots & \vdots & & \vdots \\ 0 & \bullet & \cdots & \bullet \end{array}\right).$$

2. On note $\boldsymbol{u} \in \mathbb{K}^{m+n}$, le vecteur défini par $u_i = 0$, $\forall i \in [\![1, m]\!]$ et $u_{m+i} = \underline{u_i}$, $\forall i \in [\![1, n]\!]$. Montrer que

$$\mathbb{H}(\boldsymbol{u})\mathbb{B} = \left(\begin{array}{c|c} \mathbb{B}_{1,1} & \mathbb{B}_{1,2} \\ \hline \mathbb{B}_{2,1} & \underline{\mathbb{H}}\mathbb{S} \end{array}\right).$$

Soient $k \in [\![0, n-1]\!]$ et $\mathbb{A}^{[k]} \in \mathcal{M}_n(\mathbb{K})$ la matrice bloc définie par

$$\mathbb{A}^{[k]} = \left(\begin{array}{c|c} \mathbb{R}^{[k]} & \mathbb{F}^{[k]} \\ \hline \mathbb{O} & \underline{\mathbb{A}^{[k]}} \end{array}\right)$$

où $\mathbb{R}^{[k]}$ est une matrice triangulaire supérieure d'ordre k et $\underline{\mathbb{A}^{[k]}}$ une matrice d'ordre $n-k$.

Q. 2 *1. Sous certaines hypothèses, montrer qu'il existe une matrice de Householder $\mathbb{H}^{[k+1]}$ telle que $\mathbb{H}^{[k+1]}\mathbb{A}^{[k]} = \mathbb{A}^{[k+1]}$.*

2. Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$. Montrer qu'il existe une matrice unitaire $\mathbb{Q} \in \mathcal{M}_n(\mathbb{K})$, produit d'au plus $n-1$ matrices de Housholder, et une matrice triangulaire supérieure $\mathbb{R}$ telles que $\mathbb{A} = \mathbb{Q}\mathbb{R}$.

3. Montrer que si $\mathbb{A}$ est réelle alors les coefficient diagonaux de $\mathbb{R}$ peuvent être choisi positif.

4. Montrer que si $\mathbb{A}$ est réelle inversible alors la factorisation $\mathbb{Q}\mathbb{R}$, avec $\mathbb{R}$ à coefficient diagonaux positifs, est unique.

Correction Exercice 4.2.6

Q. 1 1. D'après le (voir Corollaire 4.19, page 100) avec $\boldsymbol{a} = \boldsymbol{s}$, en posant $\alpha = \pm \|\boldsymbol{s}\|_2 e^{\imath \arg s_1}$ et

$$\underline{\boldsymbol{u}} = \frac{\boldsymbol{s} - \alpha \boldsymbol{e}_1^n}{\|\boldsymbol{s} - \alpha \boldsymbol{e}_1^n\|}$$

on obtient $\mathbb{H}(\underline{\boldsymbol{u}}) = \alpha \boldsymbol{e}_1^n$.
On pose $\underline{\mathbb{H}} = \mathbb{H}(\underline{\boldsymbol{u}})$. On a alors sous forme bloc

$$\underline{\mathbb{H}}\mathbb{S} = \underline{\mathbb{H}}\left(\begin{array}{c|ccc} \boldsymbol{s} & \bullet & \cdots & \bullet \\ & \vdots & & \vdots \\ & \bullet & \cdots & \bullet \end{array}\right) = \left(\begin{array}{c|ccc} \pm\alpha & \bullet & \cdots & \bullet \\ \hline 0 & \bullet & \cdots & \bullet \\ \vdots & \vdots & & \vdots \\ 0 & \bullet & \cdots & \bullet \end{array}\right)$$

2. On a $\boldsymbol{u} = \begin{pmatrix} \mathbf{0}_m \\ \hline \underline{\boldsymbol{u}} \end{pmatrix}$ et

$$\begin{aligned}\mathbb{H}(\boldsymbol{u}) = \mathbb{I} - 2\boldsymbol{u}\boldsymbol{u}^* &= \left(\begin{array}{c|c} \mathbb{I}_m & \mathbb{O}_{m,n} \\ \hline \mathbb{O}_{n,m} & \mathbb{I}_n \end{array}\right) - 2\left(\begin{array}{c} \mathbf{0}_m \\ \hline \underline{\boldsymbol{u}} \end{array}\right)\left(\begin{array}{c|c} \mathbf{0}_m^* & \underline{\boldsymbol{u}}^* \end{array}\right) \\ &= \left(\begin{array}{c|c} \mathbb{I}_m & \mathbb{O}_{m,n} \\ \hline \mathbb{O}_{n,m} & \mathbb{I}_n \end{array}\right) - 2\left(\begin{array}{c|c} \mathbb{O}_m & \mathbb{O}_{m,n} \\ \hline \mathbb{O}_{n,m} & \underline{\boldsymbol{u}\boldsymbol{u}}^* \end{array}\right) \\ &= \left(\begin{array}{c|c} \mathbb{I}_m & \mathbb{O}_{m,n} \\ \hline \mathbb{O}_{n,m} & \mathbb{I}_n - 2\underline{\boldsymbol{u}\boldsymbol{u}}^* \end{array}\right) = \left(\begin{array}{c|c} \mathbb{I}_m & \mathbb{O}_{m,n} \\ \hline \mathbb{O}_{n,m} & \underline{\mathbb{H}} \end{array}\right)\end{aligned}$$

Ce qui donne

$$\mathbb{H}(\boldsymbol{u})\mathbb{B} = \left(\begin{array}{c|c} \mathbb{I}_m & \mathbb{O}_{m,n} \\ \hline \mathbb{O}_{n,m} & \underline{\mathbb{H}} \end{array}\right)\left(\begin{array}{c|c} \mathbb{B}_{1,1} & \mathbb{B}_{1,2} \\ \hline \mathbb{B}_{2,1} & \mathbb{S} \end{array}\right) = \left(\begin{array}{c|c} \mathbb{B}_{1,1} & \mathbb{B}_{1,2} \\ \hline \mathbb{B}_{2,1} & \underline{\mathbb{H}}\mathbb{S} \end{array}\right).$$

Q. 2 1. On note $\underline{\boldsymbol{s}} \in \mathbb{K}^{n-k}$ le premier vecteur colonne de $\underline{\mathbb{A}^{[k]}}$. et $\boldsymbol{u} = \begin{pmatrix} \mathbf{0}_k \\ \hline \underline{\boldsymbol{s}} \end{pmatrix}$. D'après la question précédente si $\boldsymbol{s} \neq 0$ et $\boldsymbol{s}$ non colinéaire à $\boldsymbol{e}_1^{n-k}$ premier vecteur de la base canonique de $\mathbb{K}^{n-k}$ alors il existe une matrice de Householder $\mathbb{H}^{[k+1]} = \mathbb{H}(\boldsymbol{u})$ et $\alpha \in \mathbb{K}^*$ tels que

$$\mathbb{A}^{[k+1]} \overset{\text{def}}{=} \mathbb{H}^{[k+1]}\mathbb{A}^{[k]} = \left(\begin{array}{c|c} \mathbb{R}^{[k]} & \mathbb{F}^{[k]} \\ \hline \mathbb{O} & \left(\begin{array}{c|ccc} \pm\alpha & \bullet & \cdots & \bullet \\ \hline 0 & \bullet & \cdots & \bullet \\ \vdots & \vdots & & \vdots \\ 0 & \bullet & \cdots & \bullet \end{array}\right) \end{array}\right) = \left(\begin{array}{c|c} \mathbb{R}^{[k+1]} & \mathbb{F}^{[k+1]} \\ \hline \mathbb{O} & \underline{\mathbb{A}^{[k+1]}} \end{array}\right)$$

On peut remarquer que si $\boldsymbol{s} = 0$ ou $\boldsymbol{s}$ colinéaire à $\boldsymbol{e}_1^{n-k}$ alors $\mathbb{A}^{[k]}$ est déjà sous la forme $\mathbb{A}^{[k+1]}$ et donc $\mathbb{H}^{[k]} = \mathbb{I}$.

2. il suffit d'appliquer itérativement le résultat précédent $n-1$ fois en posant $\mathbb{A}^{[0]} = \mathbb{A}$ et $\mathbb{A}^{[k+1]} = \mathbb{H}^{[k+1]}\mathbb{A}^{[k]}$ où $\mathbb{H}^{[k+1]}$ est soit une matrice de Householder soit la matrice identité. Par construction la matrice $\mathbb{A}^{[n-1]}$ est triangulaire supérieure et l'on a

$$\mathbb{A}^{[n-1]} = \mathbb{H}^{[n-1]} \times \cdots \times \mathbb{H}^{[1]}\mathbb{A}$$

On pose $\mathbb{H} = \mathbb{H}^{[n-1]} \times \cdots \times \mathbb{H}^{[1]}$ et $\mathbb{R} = \mathbb{A}^{[n-1]}$. La matrice $\mathbb{H}$ est unitaire car produit de matrices unitaires. On note $\mathbb{Q} = \mathbb{H}^*$ On a

$$\mathbb{Q} = \mathbb{H}^{[1]} \times \cdots \times \mathbb{H}^{[n-1]}$$

car les matrices de Householder et matrice identité sont unitaires et hermitiennes.

3. Si $\mathbb{A}$ est réelle alors par construction $\mathbb{Q}$ et $\mathbb{R}$ sont réelles. Les coefficients diagonaux peuvent alors être choisi positif lors de la construction de chaque matrice de Householder.

4. Pour montrer l'unicité d'une telle factorisation, on note $\mathbb{Q}_1$, $\mathbb{Q}_2$, deux matrices orthogonales et $\mathbb{R}_1$, $\mathbb{R}_2$, deux matrices triangulaires à coefficients diagonaux positifs telles que

$$\mathbb{A} = \mathbb{Q}_1\mathbb{R}_1 = \mathbb{Q}_2\mathbb{R}_2.$$

Comme $\mathbb{A}$ est inversible les coefficients diagonaux de $\mathbb{R}_1$ et $\mathbb{R}_2$ sont strictement positifs. On a alors

$$\mathbb{I} = \mathbb{A}\mathbb{A}^{-1} = \mathbb{Q}_1\mathbb{R}_1\mathbb{R}_2^{-1}\mathbb{Q}_2^{-1}$$

et donc

$$\mathbb{Q}_1^{-1}\mathbb{Q}_2 = \mathbb{R}_1\mathbb{R}_2^{-1} \overset{\text{def}}{=} \mathbb{T}.$$

Comme $\mathbb{Q}_1$ est orthogonale on a $\mathbb{T} = \mathbb{Q}_1^{\mathrm{t}}\mathbb{Q}_2$ et

$$\mathbb{T}^{\mathrm{t}}\mathbb{T} = (\mathbb{Q}_1^{\mathrm{t}}\mathbb{Q}_2)^{\mathrm{t}}\mathbb{Q}_1^{\mathrm{t}}\mathbb{Q}_2 = \mathbb{Q}_2^{\mathrm{t}}\mathbb{Q}_1\mathbb{Q}_1^{\mathrm{t}}\mathbb{Q}_2 = \mathbb{I}.$$

De plus la matrice $\mathbb{T} = \mathbb{R}_1\mathbb{R}_2^{-1}$ est triangulaire supérieure à coefficients diagonaux strictement positifs puisque produit de triangulaire supérieure à coefficients diagonaux strictement positifs. D'après le théorème (factorisation de Cholesky) il existe une unique matrice $\mathbb{L}$ triangulaire inférieure à coefficients diagonaux strictement positifs telle que $\mathbb{L}\mathbb{L}^t = \mathbb{I}$ et Cette matrice $\mathbb{L}$ est la matrice identité. On en déduit que $\mathbb{T} = \mathbb{L}^t = \mathbb{I}$ et donc $\mathbb{Q}_1 = \mathbb{Q}_2$ et $\mathbb{R}_1 = \mathbb{R}_2$.

◇

Exercice 4.2.7: Algorithmique

{RSL:MD:QR:exo:16}

Q. 1 *Ecrire une fonction* FactQR *permettant de calculer la factorisation* $\mathbb{QR}$ *d'une matrice* $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$.
On pourra utiliser la fonction Householder *(voir Exercice 4.2.5, page 98).*
Q. 2 *Ecrire un programme permettant de tester cette fonction.*

Correction Exercice 4.2.7

Q. 1 L'objectif est de déterminer les matrices $\mathbb{Q}$, matrice unitaire, et $\mathbb{R}$ matrice triangulaire supérieure telle que $\mathbb{A} = \mathbb{QR}$.

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$.
Résultat : $\mathbb{Q}$: matrice unitaire de $\mathcal{M}_n(\mathbb{K})$.
$\mathbb{R}$: matrice triangulaire supérieure de $\mathcal{M}_n(\mathbb{K})$.

On rappelle la technique utilisée dans la correction de l'exercice 4.2.6 pour déterminer l'ensemble des matrices de Householder permettant de transformer la matrice $\mathbb{A}$ en une matrice triangulaire supérieure. On pose

$$\mathbb{A}^{[0]} = \mathbb{A}, \quad \mathbb{A}^{[k+1]} = \mathbb{H}^{[k+1]}\mathbb{A}^{[k]}, \ \forall k \in [\![0, n-2]\!]$$

où $\mathbb{H}^{[k+1]}$ est soit une matrice de Householder soit la matrice identité. Plus précisement, on note $\underline{\boldsymbol{s}} \in \mathbb{K}^{n-k}$ le vecteur composé des $n-k$ dernières composantes de la $k+1$-ème colonne de $\mathbb{A}^{[k]}$ et $\boldsymbol{a} = \begin{pmatrix} \mathbf{0}_k \\ \hline \underline{\boldsymbol{s}} \end{pmatrix}$.

- Si $\underline{s}_1 = 0$ ou $\underline{\boldsymbol{s}}$ colinéaire à $\boldsymbol{e}_1^{n-k}$ premier vecteur de la base canonique de $\mathbb{K}^{n-k}$ alors

$$\mathbb{H}^{[k+1]} = \mathbb{H}(\boldsymbol{u}).$$

  En notant $\boldsymbol{e}_{k+1}^n$ le $k+1$-ème vecteur de la base canonique de $\mathbb{K}^n$, cette matrice peut-être calculée avec la fonction Householder par

$$[\mathbb{H}^{[k+1]}, \alpha] \leftarrow \text{Householder}(\boldsymbol{a}, \boldsymbol{e}_{k+1}^n, 1)$$

- sinon $\mathbb{H}^{[k+1]} = \mathbb{I}$.

On a vu que dans ce cas $\mathbb{A}^{[n-1]}$ est triangulaire supérieure. On pose $\mathbb{H} = \mathbb{H}^{[n-1]} \times \cdots \times \mathbb{H}^{[1]}$ qui est une matrice unitaire. On a alors $\mathbb{R} = \mathbb{A}^{[n-1]} = \mathbb{H}\mathbb{A}$ et $\mathbb{Q} = \mathbb{H}^*$.

Algorithme 4.15 $\mathcal{R}_0$

1: Calculer et

Algorithme 4.15 $\mathcal{R}_1$

1: $\leftarrow$ $^{[n-1]} \times \cdots \times$ $^{[1]}$
2: $\leftarrow$ $*\,\mathbb{A}$
3: $\leftarrow$ *

Algorithme 4.15 $\mathcal{R}_1$

1: $\leftarrow$ $^{[n-1]} \times \cdots \times$ $^{[1]}$
2: $\leftarrow$ $*\,\mathbb{A}$
3: $\leftarrow$ *

Algorithme 4.15 $\mathcal{R}_2$

1: $\leftarrow$
2: $\mathbb{A}^{[0]} \leftarrow \mathbb{A}$
3: **Pour** $k \leftarrow 0$ **à** $n-2$ **faire**
4: Calculer $^{[k+1]}$ à partir de $\mathbb{A}^{[k]}$
5: $\mathbb{A}^{[k+1]} \leftarrow$ $^{[k+1]} * \mathbb{A}^{[k]}$
6: $\leftarrow$ $^{[k+1]}*$
7: **Fin Pour**
8: $\leftarrow$ $*\,\mathbb{A}$ ▷ ou $\leftarrow \mathbb{A}^{[n-1]}$
9: $\leftarrow$ *

Algorithme 4.15 $\mathcal{R}_2$

```
1:  ←
2:  Pour k ← 0 à n − 2 faire
3:      Calculer [k+1] à partir de 𝔸[k]
4:      𝔸[k+1] ←  [k+1] ∗ 𝔸[k]
5:         ←  [k+1] ∗
6:  Fin Pour
7:      ← 𝔸[n−1]
8:      ←  ∗
```

Algorithme 4.15 $\mathcal{R}_3$

```
1:  ←
2:  Pour k ← 0 à n − 2 faire
3:      a ← [0_k; 𝔸[k](k + 1 : n, k + 1)]
4:      [ [k+1], α] ← Householder(a, e^n_{k+1}, 1)
5:      𝔸[k+1] ←  [k+1] ∗ 𝔸[k]
6:         ←  [k+1]
7:  Fin Pour
8:      ← 𝔸[n−1]
9:      ←  ∗
```

Ici, l'opérateur $[\bullet;\bullet]$ est l'opérateur de concaténation de deux vecteurs.

{algo:RSL:FactQR}

Algorithme 4.15 Fonction FactQR

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$.
Résultat : $\mathbb{Q}$: matrice unitaire de $\mathcal{M}_n(\mathbb{K})$.
$\mathbb{R}$: matrice triangulaire supérieure de $\mathcal{M}_n(\mathbb{K})$.

```
1:  Fonction [ℚ, ℝ] ←  FactQR( 𝔸 )
2:      ℍ ← 𝕀
3:      ℝ ← 𝔸
4:      Pour k ← 0 à n − 2 faire
5:          a ← [0_k; ℝ(k + 1 : n, k + 1)]
6:          [𝕊, α] ← Householder(a, e^n_{k+1}, 1)
7:          ℝ ← 𝕊 ∗ ℝ
8:          ℍ ← 𝕊 ∗ ℍ
9:      Fin Pour
10:     ℚ ← ℍ*
11: Fin Fonction
```

Q. 2

◇

4.3 Méthodes itératives

{RSL:sec:MI}

Chapitre 5

Interpolation

Chapitre 6

Intégration numérique

Chapitre 7

Dérivation numérique

Chapitre A

Annexes

A.1 Analyse : rappels

Théorème A.1: Théorème des valeurs intermédiaires

Soit $f : [a, b] \subset \mathbb{R} \longrightarrow \mathbb{R}$ une application continue, alors

$$\forall y \in f([a, b]),\ \exists x \in [a, b] \text{ tel que } \ f(x) = y.$$

Corollaire A.2: Théorème de Bolzano

Soit $f : [a, b] \subset \mathbb{R} \longrightarrow \mathbb{R}$ une application continue. Si $f(a)$ et $f(b)$ ne sont pas de même signe (i.e. $f(a)f(b) < 0$) alors il existe au moins $c \in]a, b[$ tel que $f(c) = 0$.

Proposition A.3: Formule de Taylor-Lagrange

Soit $n \in N^*$ et $f \in \mathcal{C}^n([a, b])$ dont la dérivée n-ième est dérivable. Alors pour tout x, y dans $[a, b]$, $x \neq y$, il existe $\xi \in]x, y[$ tel que

$$f(x) = f(y) + \sum_{k=1}^{n} \frac{(x-y)^k}{k!} f^{(k)}(y) + \frac{(x-y)^{n+1}}{(n+1)!} f^{(n+1)}(\xi) \qquad \text{(A.1)}$$

{FormuleTaylorLagra

{FormuleTaylorLagra

Corollaire A.4: Théorème de la bijection

Si f est une fonction continue et strictement monotone sur un intervalle $[a, b]$ et à valeurs réelles, alors elle constitue une bijection entre $[a, b]$ et l'intervalle fermé dont les bornes sont $f(a)$ et $f(b)$.

{Theo:bijection}

Proof. Notons $J = f^{-1}([a,b])$ cet intervalle fermé, c'est-à-dire l'ensemble des réels compris entre $f(a)$ et $f(b)$.

- La monotonie de la fonction implique que l'image de l'intervalle $[a,b]$ est contenue dans J :
  - si f est croissante, pour tout $x \in [a,b]$ on a $f(a) \leqslant f(x) \leqslant f(b)$
  - si f est décroissante, pour tout $x \in [a,b]$ on a $f(b) \leqslant f(x) \leqslant f(a)$.
- Le fait que cette monotonie soit stricte assure que deux réels distincts ne peuvent avoir la même image, autrement dit la fonction est injective sur $[a,b]$.
- Enfin, le théorème des valeurs intermédiaires (qui s'appuie sur l'hypothèse de continuité) garantit que tout élément de J admet au moins un antécédent par f , c'est-à-dire que la fonction est surjective dans J.

□

Proposition A.5

riveeFctReciproque}

Soit f est une fonction bijective continue d'un intervalle ouvert $I \subset \mathbb{R}$ sur un intervalle ouvert $J \subset \mathbb{R}$. Si f est dérivable en $\alpha \in I$ et que $f'(\alpha) \neq 0$ alors sa réciproque f^{-1} est dérivable en $\beta = f(\alpha) \in J$ et

$$(f^{-1})'(\beta) = \frac{1}{f'(\alpha)} \quad \text{ou encore} \quad (f^{-1})'(\beta) = \frac{1}{f'(f^{-1}(\beta))}$$

Proof. On pose $g = f^{-1}$ et on écrit son taux d'accroissement :

$$\frac{g(y) - g(\beta)}{y - \beta} = \frac{x - \alpha}{f(x) - f(\alpha)}$$

avec $y = f(x)$. Cette fraction est l'inverse de $\frac{f(x)-f(\alpha)}{x-\alpha}$ qui tend vers $f'(\alpha) \neq 0$ quand x tend vers α. □

Théorème A.6: Théorème des accroissements finis

Soient a et b deux réels, $a < b$ et f une fonction continue sur l'intervalle fermé $[a,b]$, dérivable sur l'intervalle ouvert $]a,b[$. Alors il existe $\xi \in]a,b[$ tel que

$$f'(\xi) = \frac{f(b) - f(a)}{b - a}.$$

A.2 Algèbre linéaire

Toute cette partie peut être joyeusement omise par tout Homo sapiens *algebra linearis* compatible. Toutefois une lecture rapide permet de se raffraichir la mémoire.

Soit V un **espace vectoriel** de dimension finie n, sur le corps $\mathbb{R}$ des nombres réels, ou sur le corps $\mathbb{C}$ des nombres complexes. Notons plus généralement $\mathbb{K}$ le corps $\mathbb{R}$ ou $\mathbb{C}$.

A.2.1 Vecteurs

Une **base** de V est un ensemble $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_n\}$ de n **vecteurs linéairement indépendants**. Le vecteur $= \sum_{i=1}^{n} v_i \boldsymbol{e}_i$ sera représenté par le **vecteur colonne**

$$\boldsymbol{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}$$

et on désignera par $\boldsymbol{v}^{\mathrm{t}}$ et $\boldsymbol{v}^*$ les **vecteurs lignes** suivants

$$\boldsymbol{v}^{\mathrm{t}} = \begin{pmatrix} v_1 & v_2 & \cdots & v_n \end{pmatrix}, \quad \boldsymbol{v}^* = \begin{pmatrix} \overline{v_1} & \overline{v_2} & \cdots & \overline{v_n} \end{pmatrix}$$

où $\overline{\alpha}$ est le nombre **complexe conjugué** du nombre α.

Definition A.7 {labRAL:def:01}

- Le vecteur ligne $\boldsymbol{v}^{\mathrm{t}}$ est le **vecteur transposé** du vecteur colonne $\boldsymbol{v}$.
- Le vecteur ligne $\boldsymbol{v}^*$ est le **vecteur adjoint** du vecteur colonne $\boldsymbol{v}$.

Definition A.8 {labRAL:def:02}

L'application $\langle \bullet, \bullet \rangle : V \times V \to \mathbb{K}$ définie par

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \boldsymbol{u}^{\mathrm{t}}.\boldsymbol{v} = \boldsymbol{v}^{\mathrm{t}}.\boldsymbol{u} = \sum_{i=1}^{n} u_i v_i, \quad \text{si} \quad \mathbb{K} = \mathbb{R} \tag{A.2}$$ {ProduitScalaire

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \boldsymbol{u}^*.\boldsymbol{v} = \overline{\boldsymbol{v}^*.\boldsymbol{u}} = \overline{\langle \boldsymbol{v}, \boldsymbol{u} \rangle} = \sum_{i=1}^{n} \overline{u_i} v_i, \quad \text{si} \quad \mathbb{K} = \mathbb{C} \tag{A.3}$$ {ProduitScalaire

est appelée **produit scalaire** euclidien si $\mathbb{K} = \mathbb{R}$, hermitien[a] si $\mathbb{K} = \mathbb{C}$. Pour rappeler la dimension de l'espace, on écrit

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \langle \boldsymbol{u}, \boldsymbol{v} \rangle_n .$$

[a] La convention choisie pour le produit scalaire hermitien étant ici : linéarité à droite et semi-linéarité à gauche. Il est aussi possible de définir le produit scalaire hermitien par le complexe conjugué de A.3 :

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \boldsymbol{v}^*.\boldsymbol{u} = \sum_{i=1}^{n} u_i \overline{v_i}.$$

Dans ce cas le produit scalaire est une forme sesquilinéaire à droite.

Definition A.9

{labRAL:def:0

Soit V est un espace vectoriel muni d'un produit scalaire.

- ◇ Deux **vecteurs** $\boldsymbol{u}$ et $\boldsymbol{v}$ sont **orthogonaux** si $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = 0$.
- ◇ Un **vecteur** $\boldsymbol{v}$ est **orthogonal à une partie** U de V si

$$\forall \boldsymbol{u} \in U,\ \langle \boldsymbol{u}, \boldsymbol{v} \rangle = 0.$$

  On note $\boldsymbol{v} \perp U$.

- ◇ Un ensemble de vecteurs $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_k\}$ de l'espace V est dit **orthonormal** si

$$\langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle = \delta_{ij},\ \forall (i,j) \in [\![1,k]\!]^2$$

  où δ_{ij} est le **symbole de Kronecker** : $\delta_{ij} = \begin{cases} 1 & \text{si } i = j, \\ 0 & \text{si } i \neq j. \end{cases}$

1

Definition A.10

{labRAL:def:04}

Le vecteur nul est représenté par la lettre $\mathbf{0}$.

2

Definition A.11

{labRAL:def:05}

Soit $\boldsymbol{u} \in \mathbb{K}^n$ non nul. On définit l'**opérateur de projection** sur $\boldsymbol{u}$ par

$$\mathrm{proj}_{\boldsymbol{u}}(\boldsymbol{v}) = \frac{\langle \boldsymbol{u}, \boldsymbol{v} \rangle}{\langle \boldsymbol{u}, \boldsymbol{u} \rangle} \boldsymbol{u} = \frac{1}{\langle \boldsymbol{u}, \boldsymbol{u} \rangle} \boldsymbol{u}\boldsymbol{u}^* \boldsymbol{v},\ \forall \boldsymbol{v} \in \mathbb{K}^n. \tag{A.4}$$

3 La matrice $\mathbb{P}_{\boldsymbol{u}} = \boldsymbol{u}\boldsymbol{u}^*$ s'appelle la matrice de la projection orthogonale suivant le vecteur $\boldsymbol{u}$.

Proposition A.12: Procédé de Gram-Schmidt

{prop:AL:PGS}

Soit $\{\boldsymbol{v}_i\}_{i \in [\![1,n]\!]}$ une base de $\mathbb{K}^n$. On construit successivement les vecteurs $\boldsymbol{u}_i$

$$\boldsymbol{u}_i = \boldsymbol{v}_i - \sum_{k=1}^{i-1} \mathrm{proj}_{\boldsymbol{u}_k}(\boldsymbol{v}_i) = \boldsymbol{v}_i - \sum_{k=1}^{i-1} \frac{\langle \boldsymbol{u}_k, \boldsymbol{v}_i \rangle}{\langle \boldsymbol{u}_k, \boldsymbol{u}_k \rangle} \boldsymbol{u}_k,\ \forall i \in [\![1,n]\!].$$

Ils forment une **base orthogonale** de $\mathbb{K}^n$ et $\mathrm{Vect}(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_i) = \mathrm{Vect}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_i)$, $\forall i \in [\![1,n]\!]$ (voir Exercice A.3.5, page 129).
Pour construire une **base orthonormale** $\{\boldsymbol{z}_i\}_{i \in [\![1,n]\!]}$, il suffit de normaliser les vecteurs de la base orthogonale:

$$\boldsymbol{z}_i = \frac{\boldsymbol{u}_i}{\langle \boldsymbol{u}_i, \boldsymbol{u}_i \rangle^{1/2}},\quad \forall i \in [\![1,n]\!].$$

4

A.2.2 Matrices

5

Généralités

6

Une matrice à m lignes et n colonnes est appelée *matrice de type* (m,n), et on note $\mathcal{M}_{m,n}(\mathbb{K})$, ou simplement $\mathcal{M}_{m,n}$, l'espace vectoriel sur le corps $\mathbb{K}$ formé par les matrices de type (m,n) à éléments dans $\mathbb{K}$.

7 8 9

Une matrice $\mathbb{A} \in \mathcal{M}_{m,n}(\mathbb{K})$ d'éléments $a_{ij} \in \mathbb{K}$ est notée

$$\mathbb{A} = (a_{ij})_{1 \leqslant i \leqslant m,\ 1 \leqslant j \leqslant n},$$

10 11

le premier indice i correspond aux lignes et le second j aux colonnes. On désigne par $(\mathbb{A})_{ij}$ l'élément de la $i^{\text{ème}}$ ligne et de la $j^{\text{ème}}$ colonne.

{labRAL:def:06}

Definition A.13

La matrice nulle est représentée par la lettre $\mathbb{O}$.

Definition A.14 {labRAL:def:07}

- ◇ Soit une matrice $\mathbb{A} \in \mathcal{M}_{m,n}(\mathbb{C})$, on note $\mathbb{A}^* \in \mathcal{M}_{n,m}(\mathbb{C})$ la **matrice adjointe** de la matrice $\mathbb{A}$, définie de façon unique par

$$\langle \mathbb{A}\boldsymbol{u}, \boldsymbol{v} \rangle_m = \langle \boldsymbol{u}, \mathbb{A}^*\boldsymbol{v} \rangle_n , \ \forall \boldsymbol{u} \in \mathbb{C}^n, \ \forall \boldsymbol{v} \in \mathbb{C}^m$$

  qui entraine $(\mathbb{A}^*)_{ij} = \overline{a_{ji}}$.

- ◇ Soit une matrice $\mathbb{A} \in \mathcal{M}_{m,n}(\mathbb{R})$, on note $\mathbb{A}^{\mathtt{t}} \in \mathcal{M}_{n,m}(\mathbb{R})$ la **matrice transposée** de la matrice $\mathbb{A}$, définie de façon unique par

$$\langle \mathbb{A}\boldsymbol{u}, \boldsymbol{v} \rangle_m = \langle \boldsymbol{u}, \mathbb{A}^{\mathtt{t}}\boldsymbol{v} \rangle_n , \ \forall \boldsymbol{u} \in \mathbb{R}^n, \ \forall \boldsymbol{v} \in \mathbb{R}^m$$

  qui entraine $(\mathbb{A}^{\mathtt{t}})_{ij} = a_{ji}$.

Definition A.15 {labRAL:def:08}

Si $\mathbb{A} \in \mathcal{M}_{m,p}(\mathbb{K})$ et $\mathbb{B} \in \mathcal{M}_{p,n}(\mathbb{K})$, leur **produit** $\mathbb{AB} \in \mathcal{M}_{m,n}(\mathbb{K})$ est défini par

$$(\mathbb{AB})_{ij} = \sum_{k=1}^{p} a_{ik} b_{kj}, \forall i \in [\![1, m]\!], \ \forall j \in [\![1, n]\!]. \tag{A.5}$$

Exercice A.2.1: résultats à savoir

Soient $\mathbb{A} \in \mathcal{M}_{m,p}(\mathbb{K})$ et $\mathbb{B} \in \mathcal{M}_{p,n}(\mathbb{K})$, montrer que

$$(\mathbb{AB})^{\mathtt{t}} = \mathbb{B}^{\mathtt{t}}\mathbb{A}^{\mathtt{t}}, \quad \text{si } \mathbb{K} = \mathbb{R}, \tag{A.6}$$
$$(\mathbb{AB})^* = \mathbb{B}^*\mathbb{A}^*, \quad \text{si } \mathbb{K} = \mathbb{C} \tag{A.7}$$

Definition A.16 {labRAL:def:09}

Une matrice de type (n, n) est dite **matrice carrée** , ou **matrice d'ordre** n . On note

$$\mathcal{M}_n = \mathcal{M}_{n,n} \quad \text{ou} \quad \mathcal{M}_n(\mathbb{K}) = \mathcal{M}_{n,n}(\mathbb{K})$$

l'anneau des matrices carrées d'ordre n, à éléments dans le corps $\mathbb{K}$.

Les matrices considérées jusqu'a la fin de ce paragraphe sont carrées.

Definition A.17 {labRAL:def:10}

Si $\mathbb{A} \in \mathcal{M}_n$ alors les éléments $a_{ii} = (\mathbb{A})_{ii}$ sont appelés **éléments diagonaux** et les éléments $a_{ij} = (\mathbb{A})_{ij}, i \neq j$ sont appelés **éléments hors-diagonaux**.

Definition A.18

On appelle **matrice identitée** de $\mathcal{M}_n$ la matrice dont les éléments diagonaux sont tous égals à 1 et les éléments hors-diagonaux nulles. On la note $\mathbb{I}$ ou encore $\mathbb{I}_n$ et on a

{labRAL:def:1

$$(\mathbb{I})_{i,j} = \delta_{ij}, \ \forall (i,j) \in [\![1,n]\!]^2.$$

Definition A.19

{labRAL:def:12}

Une matrice $\mathbb{A} \in \mathcal{M}_n$ est **inversible** s'il existe une **unique** matrice de $\mathcal{M}_n$, notée $\mathbb{A}^{-1}$ et appelée **matrice inverse** de la matrice $\mathbb{A}$, telle que

$$\mathbb{A}\mathbb{A}^{-1} = \mathbb{A}^{-1}\mathbb{A} = \mathbb{I} \tag{A.8}$$

Dans le cas contraire, on dit que la matrice $\mathbb{A}$ est **singulière** .

Exercice A.2.2: ***résultats à savoir***

Soient $\mathbb{A}$ et $\mathbb{B}$ deux matrices de $\mathcal{M}_n$. Montrer que

$$\mathbb{A}\mathbb{B} = \mathbb{I} \ \Rightarrow \ \mathbb{B}\mathbb{A} = \mathbb{I}.$$

Que peut-on en conclure?

Exercice A.2.3: ***résultats à savoir***

Soient $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ et $\mathbb{B} \in \mathcal{M}_n(\mathbb{K})$ inversibles. Montrer que $\mathbb{A}\mathbb{B}$ inversible et

$$(\mathbb{A}^{\mathtt{t}})^{-1} = (\mathbb{A}^{-1})^{\mathtt{t}}, \ \text{si } \mathbb{K} = \mathbb{R}, \tag{A.9}$$
$$(\mathbb{A}^{*})^{-1} = (\mathbb{A}^{-1})^{*}, \ \text{si } \mathbb{K} = \mathbb{C}. \tag{A.10}$$
$$(\mathbb{A}\mathbb{B})^{-1} = \mathbb{B}^{-1}\mathbb{A}^{-1} \tag{A.11}$$
$$(\mathbb{A}^{-1})^{-1} = \mathbb{A} \tag{A.12}$$

Definition A.20

{labRAL:def:13}

Une matrice **carrée** $\mathbb{A}$ est :

- ◇ **symétrique** si $\mathbb{A}$ est réelle et $\mathbb{A} = \mathbb{A}^{\mathtt{t}}$,
- ◇ **hermitienne** si $\mathbb{A} = \mathbb{A}^{*}$,
- ◇ **normale** si $\mathbb{A}\mathbb{A}^{*} = \mathbb{A}^{*}\mathbb{A}$,
- ◇ **orthogonale** si $\mathbb{A}$ est réelle et $\mathbb{A}\mathbb{A}^{\mathtt{t}} = \mathbb{A}^{\mathtt{t}}\mathbb{A} = \mathbb{I}$,
- ◇ **unitaire** si $\mathbb{A}\mathbb{A}^{*} = \mathbb{A}^{*}\mathbb{A} = \mathbb{I}$,

Proposition A.21

- une matrice symétrique ou hermitienne est nécessairement normale.
- une matrice orthogonale (resp. unitaire) est nécessairement normale et inversible d'inverse $\mathbb{A}^{\mathtt{t}}$ (resp. $\mathbb{A}^{*}$).

{labRAL:def:14}

Definition A.22

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice **hermitienne**.

- ◇ Elle est **définie positive** si
$$\langle \mathbb{A}\boldsymbol{u}, \boldsymbol{u} \rangle > 0, \ \forall \boldsymbol{u} \in \mathbb{C}^n \backslash \{0\} \tag{A.13}$$
- ◇ Elle est **semi définie positive** si
$$\langle \mathbb{A}\boldsymbol{u}, \boldsymbol{u} \rangle \geqslant 0, \ \forall \boldsymbol{u} \in \mathbb{C}^n \backslash \{0\} \tag{A.14}$$

1

Exercice A.2.4

Soit $\mathbb{A} \in \mathcal{M}_n$. Que peut-on dire de la matrice $\mathbb{A}\mathbb{A}^*$? Et si la matrice $\mathbb{A}$ est inversible?
Proposer une technique permettant de générer une matrice semi-définie positive à partir d'une matrice aléatoire.

2

{labRAL:def:15}

Definition A.23

Soit $\mathbb{A} \in \mathcal{M}_n$. La trace d'une matrice carrée $\mathbb{A} = (a_{ij})$ est définie par

$$\mathrm{tr}\,(\mathbb{A}) = \sum_{i=1}^{n} a_{ii}.$$

3

{labRAL:def:16}

Definition A.24

Soit $\mathcal{T}_n$ le **groupe des permutations** de l'ensemble $\{1, 2, \ldots, n\}$. A tout élément $\sigma \in \mathcal{T}_n$, on associe la **matrice de permutation** de $\mathbb{P}_\sigma \in \mathcal{M}_n$ est définie par

$$(\mathbb{P}_\sigma)_{i,j} = \delta_{i\sigma(j)}.$$

4

Exercice A.2.5: ***résultats à savoir***

Montrer qu'une matrice de permutation est orthogonale.

5

{labRAL:def:17}

Definition A.25

Soient $\mathbb{A} = (a_{i,j})_{i,j=1}^n \in \mathcal{M}_n$ et $\mathcal{T}_n$ le **groupe des permutations** de l'ensemble $\{1, 2, \ldots, n\}$.. Le **déterminant** d'une matrice $\mathbb{A}$ est défini par

$$\det(\mathbb{A}) = \sum_{\sigma \in \mathcal{T}_n} \varepsilon_\sigma \prod_{j=1}^{n} a_{\sigma(j)j}$$

où ε_σ désigne la signature de la permutation σ.

6

Proposition A.26: Méthode de Laplace ou des coffacteurs

7

Soit $\mathbb{A} = (a_{i,j})_{i,j=1}^{n} \in \mathcal{M}_n$. On note $\mathbb{A}^{[i,j]} \in \mathcal{M}_{n-1}$ la matrice obtenue en supprimant la ligne i et la colonne j de $\mathbb{A}$. On a alors le **développement par rapport à la ligne** $i \in [\![1, n]\!]$

$$\det(\mathbb{A}) = \sum_{j=1}^{n} (-1)^{i+j} a_{i,j} \det\left(\mathbb{A}^{[i,j]}\right), \tag{A.15}$$

et le **développement par rapport à la colonne** $j \in [\![1, n]\!]$

$$\det(\mathbb{A}) = \sum_{i=1}^{n} (-1)^{i+j} a_{i,j} \det\left(\mathbb{A}^{[i,j]}\right). \tag{A.16}$$

Le terme $(-1)^{i+j} \det\left(\mathbb{A}^{[i,j]}\right)$ est appellé le **cofacteur** du terme $a_{i,j}$.

Definition A.27

{labRAL:def:18}

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$. On dit que $\lambda \in \mathbb{C}$ est **valeur propre** de $\mathbb{A}$ s'il existe $\boldsymbol{u} \in \mathbb{C}^n$ non nul tel que

$$\mathbb{A}\boldsymbol{u} = \lambda \boldsymbol{u}. \tag{A.17}$$

Le vecteur $\boldsymbol{u}$ est appelé **vecteur propre** associé à la valeur propre λ.
Le couple $(\lambda, \boldsymbol{u})$ est appelé **élément propre** de $\mathbb{A}$.

Definition A.28

{labRAL:def:19}

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$. Soit $\lambda \in \mathbb{C}$ une valeur propre de $\mathbb{A}$. Le sous-espace

$$E_\lambda = \{\boldsymbol{u} \in \mathbb{C}^n : \mathbb{A}\boldsymbol{u} = \lambda \boldsymbol{u}\} = \ker(\mathbb{A} - \lambda \mathbb{I}) \tag{A.18}$$

est appelé **sous-espace propre** associé à la valeur propre λ.

Definition A.29

{labRAL:def:20}

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$. Le polynôme de degré n défini par

$$\mathcal{P}_{\mathbb{A}}(\lambda) = \det(\mathbb{A} - \lambda \mathbb{I}) \tag{A.19}$$

est appellé **polynôme caractéristique** de la matrice $\mathbb{A}$.

Proposition A.30

{AL:prop:100}

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$.

- ◇ Les racines complexes du polynôme caractéristique $\mathcal{P}_{\mathbb{A}}$ sont les valeurs propres de la matrice $\mathbb{A}$.
- ◇ Si la racine λ de $\mathcal{P}_{\mathbb{A}}$ est de multiplicité k, on dit que la valeur propre λ est de multiplicité algébrique k.
- ◇ La matrice $\mathbb{A}$ possède n valeurs propres distinctes ou non.

Proposition A.31

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$. Soit $\lambda_1, \ldots, \lambda_k$ des valeurs propres distinctes de $\mathbb{A}$. On a

$$E_{\lambda_1} + \ldots + E_{\lambda_k} = E_{\lambda_1} \oplus \ldots \oplus E_{\lambda_k}. \tag{A.20}$$

Definition A.32

{labRAL:def:21}

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$. On note $\lambda_i(\mathbb{A})$, $i \in [\![1, n]\!]$, les n valeurs propres de $\mathbb{A}$. Le **spectre** de la matrice $\mathbb{A}$ est le sous-ensemble

$$\mathrm{Sp}(\mathbb{A}) = \bigcup_{i=1}^{n} \{\lambda_i(\mathbb{A})\} \tag{A.21}$$

du plan complexe.

Proposition A.33

Soient $\mathbb{A} \in \mathcal{M}_n$ et $\mathbb{B} \in \mathcal{M}_n$. On a les relations suivantes

$$\begin{aligned}
\mathrm{tr}(\mathbb{A}) &= \sum_{i=1}^{n} \lambda_i(\mathbb{A}), && \text{(A.22)}\\
\det(\mathbb{A}) &= \prod_{i=1}^{n} \lambda_i(\mathbb{A}), && \text{(A.23)}\\
\mathrm{tr}(\mathbb{AB}) &= \mathrm{tr}(\mathbb{BA}), && \text{(A.24)}\\
\mathrm{tr}(\mathbb{A}+\mathbb{B}) &= \mathrm{tr}\,A + \mathrm{tr}\,B, && \text{(A.25)}\\
\det(\mathbb{AB}) &= \det(\mathbb{A})\det(\mathbb{B}) = \det(\mathbb{BA}), && \text{(A.26)}\\
\det(\mathbb{A}^*) &= \overline{\det(\mathbb{A})}. && \text{(A.27)}
\end{aligned}$$

Definition A.34

{labRAL:def:22}

Le **rayon spectral** d'une matrice $\mathbb{A} \in \mathcal{M}_n$ est le nombre $\geqslant 0$ défini par

$$\rho(\mathbb{A}) = \max\{|\lambda_i(\mathbb{A})|\,;\, i \in [\![1, n]\!]\}$$

Definition A.35

{labRAL:def:23}

Soit X et Y deux espaces vectoriels

- ◇ On note $\ker(\mathbb{A}) = \{\boldsymbol{v} \in X\,;\, \mathbb{A}\boldsymbol{v} = 0\}$ le **noyau** de l'application linéaire $\mathbb{A} : X \to Y$.
- ◇ On note $\mathrm{im}(\mathbb{A}) = \{\mathbb{A}\boldsymbol{v} \in Y\,;\, \boldsymbol{v} \in X\}$ l'**image** de l'application linéaire $\mathbb{A} : X \to Y$.

Definition A.36

{labRAL:def:24}

Le rang d'une matrice $\mathbb{A}$, noté $\mathrm{rang}(\mathbb{A})$, est défini comme le rang de l'application linéaire $\mathbb{A}$ qui lui et associé

$$\mathrm{rang}(\mathbb{A}) = \dim(\mathrm{im}(\mathbb{A})).$$

Matrices particulières

Definition A.37 {labRAL:def:2

Une matrice carrée $\mathbb{A} \in \mathcal{M}_n$ est :

- ◇ **diagonale** si $a_{ij} = 0$ pour $i \neq j$,
- ◇ **triangulaire supérieure** si $a_{ij} = 0$ pour $i > j$,
- ◇ **triangulaire inférieure** si $a_{ij} = 0$ pour $i < j$,
- ◇ **triangulaire** si elle est triangulaire supérieure ou triangulaire inférieure
- ◇ **à diagonale dominante** si

$$|a_{ii}| \geqslant \sum_{j \neq i} |a_{ij}|, \ \forall i \in [\![1, n]\!], \tag{A.28}$$

- ◇ **à diagonale strictement dominante** si

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}|, \ \forall i \in [\![1, n]\!]. \tag{A.29}$$

1

Proposition A.38 {AL:prop:99}

Soient $\mathbb{A}$ et $\mathbb{B}$ deux matrices de $\mathcal{M}_n(\mathbb{K})$ triangulaires inférieures (resp. triangulaires supérieures). Alors la matrice $\mathbb{AB}$ est aussi triangulaire inférieure (resp. triangulaire supérieure).
De plus on a

$$(\mathbb{AB})_{i,i} = A_{i,i} B_{i,i}, \ \forall i \in [\![1, n]\!].$$

2

3 *Proof.* (voir Exercice A.3.2, page 128) □

Proposition A.39 {AL:prop:100}

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ une matrice triangulaire inférieure (resp. triangulaire supérieure).

1. $\mathbb{A}$ est inversible si et seulement si ses éléments diagonaux sont tous non nuls (i.e. $A_{i,i} \neq 0$, $\forall i \in [\![1, n]\!]$).
2. Si $\mathbb{A}$ est inversible alors son inverse est triangulaire inférieure (resp. triangulaire supérieure) et

$$(\mathbb{A}^{-1})_{i,i} = \frac{1}{(\mathbb{A})_{i,i}}$$

4

5 *Proof.* (voir Exercice A.3.10, page 131) □

Definition A.40 {labRAL:def:30}

On appelle **matrice bande** une matrice $\mathbb{A}$ telle que $a_{ij} \neq 0$ pour $|j - i| \leqslant c$. c est la **demi largeur de bande**.

Lorsque $c = 1$, la matrice est dite **tridiagonale** . Lorsque $c = 2$, la matrice est dite **pentadiagonale** .

6

7 **Definition A.41**

abRAL:def:26}

On appelle **sous-matrice** d'une matrice donnée, la matrice obtenue en supprimant certaines lignes et certaines colonnes. En particulier, si on supprime les $(n-k)$ dernières lignes et colonnes d'une matrice carrée $\mathbb{A}$ d'ordre n, on obtient la **sous matrice principale** d'ordre k.

1

Definition A.42

{labRAL:def:27}

On appelle **matrice bloc** une matrice $\mathbb{A} \in \mathcal{M}_{N,M}$ écrite sous la forme

$$\mathbb{A} = \left(\begin{array}{c|c|c} \mathbb{A}_{1,1} & \cdots & \mathbb{A}_{1,q} \\ \hline \vdots & & \vdots \\ \hline \mathbb{A}_{p,1} & \cdots & \mathbb{A}_{p,q} \end{array}\right)$$

où $\forall i \in [\![1,p]\!]$, $\forall j \in [\![1,q]\!]$, $\mathbb{A}_{i,j}$ est une matrice de $\mathcal{M}_{n_i,m_j}$. On a $N = \sum_{i=1}^{p} n_i$ et $M = \sum_{j=1}^{q} m_j$.

On dit que $\mathbb{A}$ et une matrice **bloc-carrée** si $p = q$ et si tous les blocs diagonaux sont des matrices carrées.

2

Propriété A.43: Multiplication de matrices blocs

Soient $\mathbb{A} \in \mathcal{M}_{N,M}$ et $\mathbb{B} \in \mathcal{M}_{M,S}$. Le produit $\mathbb{P} = \mathbb{A}\mathbb{B} \in \mathcal{M}_{N,S}$ peut s'écrire sous forme bloc si les matrices $\mathbb{A}$ et $\mathbb{B}$ sont *compatibles par blocs* : il faut que le nombre de blocs colonne de $\mathbb{A}$ soit égale au nombre de blocs ligne de $\mathbb{B}$ avec correspondance des dimensions.

$$\mathbb{A} = \left(\begin{array}{c|c|c} \mathbb{A}_{1,1} & \cdots & \mathbb{A}_{1,q} \\ \hline \vdots & & \vdots \\ \hline \mathbb{A}_{p,1} & \cdots & \mathbb{A}_{p,q} \end{array}\right) \text{ et } \mathbb{B} = \left(\begin{array}{c|c|c} \mathbb{B}_{1,1} & \cdots & \mathbb{A}_{1,r} \\ \hline \vdots & & \vdots \\ \hline \mathbb{B}_{q,1} & \cdots & \mathbb{A}_{q,r} \end{array}\right)$$

avec $\mathbb{A}_{i,k} \in \mathcal{M}_{n_i,m_k}$ et $\mathbb{B}_{k,j} \in \mathcal{M}_{m_k,s_j}$ pour tout $i \in [\![1,p]\!]$, $k \in [\![1,q]\!]$ et $j \in [\![1,r]\!]$. La matrice produit $\mathbb{P}$ s'écrit alors sous la forme bloc

$$\mathbb{P} = \left(\begin{array}{c|c|c} \mathbb{P}_{1,1} & \cdots & \mathbb{P}_{1,r} \\ \hline \vdots & & \vdots \\ \hline \mathbb{P}_{p,1} & \cdots & \mathbb{P}_{p,r} \end{array}\right)$$

avec $\forall i \in [\![1,p]\!]$, $\forall j \in [\![1,r]\!]$ $\mathbb{P}_{i,j} \in \mathcal{M}_{n_i,s_j}$ et

$$\mathbb{P}_{i,j} = \sum_{k=1}^{q} \mathbb{A}_{i,k}\mathbb{B}_{k,j}.$$

3

Definition A.44

{labRAL:def:28}

On dit qu'une matrice bloc-carrée $\mathbb{A}$ est **triangulaire inférieure** (resp. **supérieure**) **par blocs** si elle peut s'écrire sous la forme d'une matrice bloc avec les sous matrices $\mathbb{A}_{i,j} = 0$ pour $i < j$ (resp. $i > j$). . Elle s'écrit donc sous la forme

$$\mathbb{A} = \left(\begin{array}{c|c|c|c} \mathbb{A}_{1,1} & \mathbb{0} & \cdots & \mathbb{0} \\ \hline \vdots & \ddots & \ddots & \vdots \\ \hline \vdots & \ddots & \ddots & \mathbb{0} \\ \hline \mathbb{A}_{n,1} & \cdots & \cdots & \mathbb{A}_{n,n} \end{array}\right) \text{ (resp. } \mathbb{A} = \left(\begin{array}{c|c|c|c} \mathbb{A}_{1,1} & \cdots & \cdots & \mathbb{A}_{n,1} \\ \hline \mathbb{0} & \ddots & \ddots & \vdots \\ \hline \vdots & \ddots & \ddots & \vdots \\ \hline \mathbb{0} & \cdots & \mathbb{0} & \mathbb{A}_{n,n} \end{array}\right)\text{).}$$

4

Definition A.45 {labRAL:def:2

On dit qu'une matrice bloc-carrée $\mathbb{A}$ est **diagonale par blocs** ou **bloc-diagonale** si elle peut s'écrire sous la forme d'une matrice bloc avec les sous matrices $\mathbb{A}_{i,j} = 0$ pour $i \neq j$. Elle s'écrit donc sous la forme

$$\mathbb{A} = \left(\begin{array}{c|c|c|c} \mathbb{A}_{1,1} & \mathbb{0} & \cdots & \mathbb{0} \\ \hline \mathbb{0} & \ddots & \ddots & \vdots \\ \hline \vdots & \ddots & \ddots & \mathbb{0} \\ \hline \mathbb{0} & \cdots & \mathbb{0} & \mathbb{A}_{n,n} \end{array}\right)$$

1

Proposition A.46

{labRAL:prop:29}

Soit $\mathbb{A}$ une matrice bloc-carré décomposée en $n \times n$ blocs. Si $\mathbb{A}$ est **bloc-diagonale** ou **triangulaire par blocs** alors son déterminant est le produit des déterminant des blocs diagonaux :

$$\det \mathbb{A} = \prod_{i=1}^{n} \det \mathbb{A}_{i,i} \tag{A.30}$$

2

Proposition A.47

{labRAL:prop:30}

Soit $\mathbb{A}$ une matrice bloc-carré **inversible** décomposée en $n \times n$ blocs.

- Si $\mathbb{A}$ est **bloc-diagonale** alors son inverse (décomposée en $n \times n$ blocs) est aussi **bloc-diagonale**.
- Si $\mathbb{A}$ est **triangulaire inférieure par blocs** (resp. supérieure) alors son inverse (décomposée en $n \times n$ blocs) est aussi **triangulaire inférieure par blocs** (resp. supérieure).

Dans ces deux cas les blocs diagonaux de la matrice inverse sont les inverses des blocs diagonaux de $\mathbb{A}$. On a donc

$$\mathbb{A} = \left(\begin{array}{c|c|c|c} \mathbb{A}_{1,1} & \mathbb{0} & \cdots & \mathbb{0} \\ \hline \mathbb{0} & \ddots & \ddots & \vdots \\ \hline \vdots & \ddots & \ddots & \mathbb{0} \\ \hline \mathbb{0} & \cdots & \circ & \mathbb{A}_{n,n} \end{array}\right) \text{ et } \mathbb{A}^{-1} = \left(\begin{array}{c|c|c|c} \mathbb{A}_{1,1}^{-1} & \mathbb{0} & \cdots & \mathbb{0} \\ \hline \mathbb{0} & \ddots & \ddots & \vdots \\ \hline \vdots & \ddots & \ddots & \mathbb{0} \\ \hline \mathbb{0} & \cdots & \mathbb{0} & \mathbb{A}_{n,n}^{-1} \end{array}\right)$$

$$\mathbb{A} = \left(\begin{array}{c|c|c|c} \mathbb{A}_{1,1} & \mathbb{0} & \cdots & \mathbb{0} \\ \hline \vdots & \ddots & \ddots & \vdots \\ \hline \vdots & \ddots & \ddots & \mathbb{0} \\ \hline \mathbb{A}_{n,1} & \cdots & \cdots & \mathbb{A}_{n,n} \end{array}\right) \text{ et } \mathbb{A}^{-1} = \left(\begin{array}{c|c|c|c} \mathbb{A}_{1,1}^{-1} & \mathbb{0} & \cdots & \mathbb{0} \\ \hline \bullet & \ddots & \ddots & \vdots \\ \hline \vdots & \ddots & \ddots & \mathbb{0} \\ \hline \bullet & \cdots & \bullet & \mathbb{A}_{n,n}^{-1} \end{array}\right)$$

$$\mathbb{A} = \left(\begin{array}{c|c|c|c} \mathbb{A}_{1,1} & \cdots & \cdots & \mathbb{A}_{n,1} \\ \hline \mathbb{0} & \ddots & \ddots & \vdots \\ \hline \vdots & \ddots & \ddots & \vdots \\ \hline \mathbb{0} & \cdots & \mathbb{0} & \mathbb{A}_{n,n} \end{array}\right) \text{ et } \mathbb{A}^{-1} = \left(\begin{array}{c|c|c|c} \mathbb{A}_{1,1}^{-1} & \bullet & \cdots & \bullet \\ \hline \mathbb{0} & \ddots & \ddots & \vdots \\ \hline \vdots & \ddots & \ddots & \bullet \\ \hline \mathbb{0} & \cdots & \mathbb{0} & \mathbb{A}_{n,n}^{-1} \end{array}\right)$$

3

A.2.3 Normes vectorielles et normes matricielles

4

Definition A.48

Une **norme** sur un espace vectoriel V est une application $\|\bullet\| : V \to \mathbb{R}^+$ qui vérifie les propriétés suivantes

- $\|\boldsymbol{v}\| = 0 \Longleftrightarrow \boldsymbol{v} = 0$,
- $\|\alpha\boldsymbol{v}\| = |\alpha| \|\boldsymbol{v}\|, \ \forall \alpha \in \mathbb{K}, \ \forall \boldsymbol{v} \in V$,
- $\|\boldsymbol{u}+\boldsymbol{v}\| \leqslant \|\boldsymbol{u}\| + \|\boldsymbol{v}\|, \ \forall (\boldsymbol{u},\boldsymbol{v}) \in V^2$ (inégalité triangulaire).

Une norme sur V est également appelée **norme vectorielle** . On appelle **espace vectoriel normé** un espace vectoriel muni d'une norme.

Les trois normes suivantes sont les plus couramment utilisées :

$$\|\boldsymbol{v}\|_1 = \sum_{i=1}^{n} |v_i|$$
$$\|\boldsymbol{v}\|_2 = \left(\sum_{i=1}^{n} |v_i|^2 \right)^{1/2}$$
$$\|\boldsymbol{v}\|_\infty = \max_i |v_i| .$$

Théorème A.49

Soit V un espace de dimension finie. Pour tout nombre réel $p \geqslant 1$, l'application $\|\bullet\|_p$ définie par

$$\|\boldsymbol{v}\|_p = \left(\sum_{i=1}^{n} |v_i|^p \right)^{1/p}$$

est une norme.

Claim A.50 Pour $p > 1$ et $\frac{1}{p} + \frac{1}{q} = 1$, on a $\forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{K}^n$

$$\|\boldsymbol{u}\boldsymbol{v}\|_1 = \sum_{i=1}^{n} |u_i v_i| \leqslant \left(\sum_{i=1}^{n} |u_i|^p \right)^{1/p} \left(\sum_{i=1}^{n} |v_i|^q \right)^{1/q} = \|\boldsymbol{u}\|_p \|\boldsymbol{v}\|_q . \tag{A.31}$$

Cette inégalité s'appelle l'**inégalité de Hölder**.

Definition A.51

Deux **normes** $\|\bullet\|$ et $\|\bullet\|'$, définies sur un même espace vectoriel V, sont **équivalentes** s'il exite deux constantes C et C' telles que

$$\|\boldsymbol{v}\|' \leqslant C \|\boldsymbol{v}\| \quad \text{et} \quad \|\boldsymbol{v}\| \leqslant C' \|\boldsymbol{v}\|' \quad \text{pour tout } \boldsymbol{v} \in V. \tag{A.32}$$

Claim A.52 Sur un espace vectoriel de dimension finie toutes les normes sont équivalentes.

Definition A.53

Une **norme matricielle** sur $\mathcal{M}_n(\mathbb{K})$ est une application $\|\bullet\| : \mathcal{M}_n(\mathbb{K}) \to \mathbb{R}^+$ vérifiant

1. $\|\mathbb{A}\| = 0 \Longleftrightarrow \mathbb{A} = 0$,
2. $\|\alpha\mathbb{A}\| = |\alpha| \, \|\mathbb{A}\|$, $\forall \alpha \in \mathbb{K}$, $\forall \mathbb{A} \in \mathcal{M}_n(\mathbb{K})$,
3. $\|\mathbb{A} + \mathbb{B}\| \leqslant \|\mathbb{A}\| + \|\mathbb{B}\|$, $\forall (\mathbb{A}, \mathbb{B}) \in \mathcal{M}_n(\mathbb{K})^2$ (inégalité triangulaire)
4. $\|\mathbb{A}\mathbb{B}\| \leqslant \|\mathbb{A}\| \, \|\mathbb{B}\|$, $\forall (\mathbb{A}, \mathbb{B}) \in \mathcal{M}_n(\mathbb{K})^2$

Claim A.54 Etant donné une norme vectorielle $\|\bullet\|$ sur $\mathbb{K}^n$, l'application $\|\bullet\|_s : \mathcal{M}_n(\mathbb{K}) \to \mathbb{R}^+$ définie par

$$\|\mathbb{A}\|_s = \sup_{\substack{\boldsymbol{v} \in \mathbb{K}^n \\ \boldsymbol{v} \neq 0}} \frac{\|\mathbb{A}\boldsymbol{v}\|}{\|\boldsymbol{v}\|} = \sup_{\substack{\boldsymbol{v} \in \mathbb{K}^n \\ \|\boldsymbol{v}\| \leqslant 1}} \|\mathbb{A}\boldsymbol{v}\| = \sup_{\substack{\boldsymbol{v} \in \mathbb{K}^n \\ \|\boldsymbol{v}\| = 1}} \|\mathbb{A}\boldsymbol{v}\| , \tag{A.33}$$

est une norme matricielle, appelée **norme matricielle subordonnée** (à la norme vectorielle donnée).
De plus

$$\|\mathbb{A}\boldsymbol{v}\| \leqslant \|\mathbb{A}\|_s \, \|\boldsymbol{v}\| \quad \forall \boldsymbol{v} \in \mathbb{K}^n \tag{A.34}$$

et la norme $\|\mathbb{A}\|$ peut se définir aussi par

$$\|\mathbb{A}\|_s = \inf \{\alpha \in \mathbb{R} : \|\mathbb{A}\boldsymbol{v}\| \leqslant \alpha \|\boldsymbol{v}\| , \ \forall \boldsymbol{v} \in \mathbb{K}^n\} . \tag{A.35}$$

Il existe au moins un vecteur $\boldsymbol{u} \in \mathbb{K}^n$ tel que

$$\boldsymbol{u} \neq 0 \ \text{ et } \ \|\mathbb{A}\boldsymbol{u}\| = \|\mathbb{A}\|_s \, \|\boldsymbol{u}\| . \tag{A.36}$$

Enfin une norme subordonnée vérifie toujours

$$\|\mathbb{I}\|_s = 1 \tag{A.37}$$

Théorème A.55

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$. On a

$$\|\mathbb{A}\|_1 \overset{\text{déf.}}{=} \sup_{\substack{\boldsymbol{v} \in \mathbb{C}^n \\ \boldsymbol{v} \neq 0}} \frac{\|\mathbb{A}\boldsymbol{v}\|_1}{\|\boldsymbol{v}\|_1} = \max_{j \in [\![1,n]\!]} \sum_{i=1}^{n} |a_{ij}| \tag{A.38}$$

$$\|\mathbb{A}\|_2 \overset{\text{déf.}}{=} \sup_{\substack{\boldsymbol{v} \in \mathbb{C}^n \\ \boldsymbol{v} \neq 0}} \frac{\|\mathbb{A}\boldsymbol{v}\|_2}{\|\boldsymbol{v}\|_2} = \sqrt{\rho\left(\mathbb{A}^*\mathbb{A}\right)} = \sqrt{\rho\left(\mathbb{A}\mathbb{A}^*\right)} = \|\mathbb{A}^*\|_2 \tag{A.39}$$

$$\|\mathbb{A}\|_\infty \overset{\text{déf.}}{=} \sup_{\substack{\boldsymbol{v} \in \mathbb{C}^n \\ \boldsymbol{v} \neq 0}} \frac{\|\mathbb{A}\boldsymbol{v}\|_\infty}{\|\boldsymbol{v}\|_\infty} = \max_{i \in [\![1,n]\!]} \sum_{j=1}^{n} |a_{ij}| \tag{A.40}$$

La norme $\|\bullet\|_2$ est invariante par transformation unitaire :

$$\mathbb{U}\mathbb{U}^* = \mathbb{I} \Longrightarrow \|\mathbb{A}\|_2 = \|\mathbb{A}\mathbb{U}\|_2 = \|\mathbb{U}\mathbb{A}\|_2 = \|\mathbb{U}^*\mathbb{A}\mathbb{U}\|_2 . \tag{A.41}$$

Par ailleurs, si la matrice $\mathbb{A}$ est normale :

$$\mathbb{A}\mathbb{A}^* = \mathbb{A}^*\mathbb{A} \Longrightarrow \|\mathbb{A}\|_2 = \rho(\mathbb{A}). \tag{A.42}$$

Remarque A.56
1. Si une matrice $\mathbb{A}$ est hermitienne, ou symétrique (donc normale), on a $\|\mathbb{A}\|_2 = \rho(\mathbb{A})$.
2. Si une matrice $\mathbb{A}$ est unitaire, ou orthogonale (donc normale), on a $\|\mathbb{A}\|_2 = 1$.

Théorème A.57

1. Soit $\mathbb{A}$ une matrice carrée quelconque et $\|\bullet\|$ une norme matricielle subordonnée ou non, quelconque. Alors
$$\rho(\mathbb{A}) \leqslant \|\mathbb{A}\|. \tag{A.43}$$
2. Etant donné une matrice $\mathbb{A}$ et un nombre $\varepsilon > 0$, il existe au moins une norme matricielle subordonnée telle que
$$\|\mathbb{A}\| \leqslant \rho(\mathbb{A}) + \varepsilon. \tag{A.44}$$

Théorème A.58

L'application $\|\bullet\|_E : \mathcal{M}_n \to \mathbb{R}^+$ définie par

$$\|\mathbb{A}\|_E = \left(\sum_{(i,j)=\in[\![1,n]\!]^2} |a_{ij}|^2 \right)^{1/2} = \sqrt{\mathrm{tr}\,(\mathbb{A}^*\mathbb{A})}, \tag{A.45}$$

pour toute matrice $\mathbb{A} = (a_{ij})$ d'ordre n, est une norme matricielle non subordonnée (pour $n \geqslant 2$), invariante par transformation unitaire et qui vérifie

$$\|\mathbb{A}\|_2 \leqslant \|\mathbb{A}\|_E \leqslant \sqrt{n}\,\|\mathbb{A}\|_2\,, \ \forall \mathbb{A} \in \mathcal{M}_n. \tag{A.46}$$

De plus $\|\mathbb{I}\|_E = \sqrt{n}$.

Théorème A.59

1. Soit $\|\bullet\|$ une norme matricielle subordonnée, et $\mathbb{B}$ une matrice vérifiant
$$\|\mathbb{B}\| < 1.$$
Alors la matrice $(\mathbb{I} + \mathbb{B})$ est inversible, et
$$\left\|(\mathbb{I} + \mathbb{B})^{-1}\right\| \leqslant \frac{1}{1 - \|\mathbb{B}\|}.$$
2. Si une matrice de la forme $(\mathbb{I} + \mathbb{B})$ est singulière, alors nécessairement
$$\|\mathbb{B}\| \geqslant 1$$
pour toute norme matricielle, subordonnée ou non.

A.2.4 Réduction des matrices

Definition A.60

Soit $A : V \to V$ une application linéaire, représenté par une matrice carrée $\mathbb{A} \in \mathcal{M}_n$ relativement à une base $\{\boldsymbol{e}_i\}_{i\in[\![1,n]\!]}$. Relativement à une autre base $\{\boldsymbol{f}_i\}_{i\in[\![1,n]\!]}$, la même application est représentée par la matrice

$$\mathbb{B} = \mathbb{P}^{-1}\mathbb{A}\mathbb{P} \tag{A.47}$$

où $\mathbb{P}$ est la matrice inversible dont le j-ème vecteur colonne est formé des composantes du vecteur $\boldsymbol{f}_j$ dans la base $\{\boldsymbol{e}_i\}_{i\in[\![1,n]\!]}$:

$$\mathbb{P} = \begin{pmatrix} \langle \boldsymbol{e}_1, \boldsymbol{f}_1\rangle & \langle \boldsymbol{e}_1, \boldsymbol{f}_2\rangle & \cdots & \langle \boldsymbol{e}_1, \boldsymbol{f}_n\rangle \\ \langle \boldsymbol{e}_2, \boldsymbol{f}_1\rangle & \langle \boldsymbol{e}_1, \boldsymbol{f}_2\rangle & \ddots & \vdots \\ \vdots & \ddots & \ddots & \langle \boldsymbol{e}_{n-1}, \boldsymbol{f}_n\rangle \\ \langle \boldsymbol{e}_n, \boldsymbol{f}_1\rangle & \cdots & \langle \boldsymbol{e}_n, \boldsymbol{f}_{n-1}\rangle & \langle \boldsymbol{e}_n, \boldsymbol{f}_n\rangle \end{pmatrix} \tag{A.48}$$

La matrice $\mathbb{P}$ est appelée **matrice de passage de la base** $\{\boldsymbol{e}_i\}_{i\in[\![1,n]\!]}$ **dans le base** $\{\boldsymbol{f}_i\}_{i\in[\![1,n]\!]}$.

Definition A.61

On dit que la matrice carrée $\mathbb{A}$ est diagonalisable s'il existe une matrice inversible $\mathbb{P}$ telle que la matrice $\mathbb{P}^{-1}\mathbb{A}\mathbb{P}$ soit diagonale.

Claim A.62 On notera que, dans le cas où $\mathbb{A} \in \mathcal{M}_n$ est diagonalisable, les éléments diagonaux de la matrice $\mathbb{P}^{-1}\mathbb{A}\mathbb{P}$ sont les valeurs propres $\lambda_1, \lambda_2, \ldots, \lambda_n$ de la matrice $\mathbb{A}$, et que le j-ème vecteur colonne $\boldsymbol{p}_j$ de la matrice $\mathbb{P}$ est formé des composantes, dans la même base que $\mathbb{A}$, d'un vecteur propre associé à la valeur propre λ_j. On a

$$\mathbb{P}^{-1}\mathbb{A}\mathbb{P} = \operatorname{diag}(\lambda_1, \ldots, \lambda_n) \Longleftrightarrow \mathbb{A}\boldsymbol{p}_j = \lambda_j \boldsymbol{p}_j, \ \forall j \in [\![1, n]\!]. \tag{A.49}$$

C'est à dire qu'une matrice est diagonalisable si, et seulement si, il existe une base de vecteurs propres.

Théorème A.63

1. Etant donnée une matrice **carrée** $\mathbb{A}$, il existe une matrice **unitaire** $\mathbb{U}$ telle que la matrice $\mathbb{U}^{-1}\mathbb{A}\mathbb{U}$ soit **triangulaire**.
2. Etant donnée une matrice **normale** $\mathbb{A}$, il existe une matrice **unitaire** $\mathbb{U}$ telle que la matrice $\mathbb{U}^{-1}\mathbb{A}\mathbb{U}$ soit **diagonale**.
3. Etant donnée une matrice **symétrique** $\mathbb{A}$, il existe une matrice **orthogonale** $\mathbb{O}$ telle que la matrice $\mathbb{O}^{-1}\mathbb{A}\mathbb{O}$ soit **diagonale**.

A.2.5 Suites de vecteurs et de matrices

Definition A.64

Soit V un espace vectoriel muni d'une norme $\|\bullet\|$, on dit qu'une suite $(\boldsymbol{v}_k)$ d'éléments de V **converge vers un élément** $\boldsymbol{v} \in V$, si

$$\lim_{k\to\infty} \|\boldsymbol{v}_k - \boldsymbol{v}\| = 0$$

et on écrit

$$\boldsymbol{v} = \lim_{k\to\infty} \boldsymbol{v}_k.$$

Théorème A.65

Soit $\mathbb{B}$ une matrice carrée. Les conditions suivantes sont équivalentes :

1. $\lim_{k\to\infty} \mathbb{B}^k = 0$,
2. $\lim_{k\to\infty} \mathbb{B}^k \boldsymbol{v} = 0$ pour tout vecteur $\boldsymbol{v}$,
3. $\rho(\mathbb{B}) < 1$,
4. $\|\mathbb{B}\| < 1$ pour au moins une norme matricielle subordonnée $\|\bullet\|$.

Théorème A.66

Soit $\mathbb{B}$ une matrice carrée, et $\|\bullet\|$ une norme matricielle quelconque. Alors

$$\lim_{k\to\infty} \left\|\mathbb{B}^k\right\|^{1/k} = \rho(\mathbb{B}).$$

A.3 Receuil d'exercices

A.3.1 Algèbre linéaire

Sur les matrices

Exercice A.3.1

Soit $\mathbb{A} \in \mathcal{M}_{m,n}(\mathbb{R})$ et $\mathbb{B} \in \mathcal{M}_{n,m}(\mathbb{R})$ telles que

$$\langle \mathbb{A}\boldsymbol{u}, \boldsymbol{v}\rangle_m = \langle \boldsymbol{u}, \mathbb{B}\boldsymbol{v}\rangle_n\,,\ \forall \boldsymbol{u} \in \mathbb{R}^n,\ \forall \boldsymbol{v} \in \mathbb{R}^m.$$

Exprimer les éléments de la matrice $\mathbb{B}$ en fonction de ceux de la matrice $\mathbb{A}$.

Exercice A.3.2

{RE:AL:matric

Soient $\mathbb{A}$ et $\mathbb{B}$ deux matrices triangulaires supérieures de $\mathcal{M}_n$. Soient $\mathbb{E}$ et $\mathbb{F}$ deux matrices triangulaires inférieures de $\mathcal{M}_n$.

Q. 1 1. *Que peut-on dire des matrices* $\mathbb{A}^*$ *et* $(\mathbb{A}^*)^*$*?*

2. *Montrer que* $\mathbb{C} = \mathbb{A}\mathbb{B}$ *est triangulaire supérieure et que* $C_{i,i} = A_{i,i}B_{i,i}$, $\forall i \in [\![1, n]\!]$.

3. *Montrer que* $\mathbb{G} = \mathbb{E}\mathbb{F}$ *est triangulaire inférieure et que* $G_{i,i} = E_{i,i}F_{i,i}$, $\forall i \in [\![1, n]\!]$.

4. *Que peut-on dire des matrices* $\mathbb{A}\mathbb{E}$ *et* $\mathbb{E}\mathbb{A}$*?*

Q. 2 1. *Calculer* $\det(\mathbb{A})$.

2. *Déterminer les valeurs propres de* $\mathbb{A}$.

3. *Que peut-on dire si les éléments diagonaux de* $\mathbb{A}$ *sont tous distincts?*

Q. 3 *Soit* $\mathbb{D}$ *la matrice définie par*

$$\mathbb{D}\begin{pmatrix} 2 & 1 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{pmatrix}.$$

1. *La matrice* $\mathbb{D}$ *est-elle inversible? Si oui calculer son inverse.*

2. *Pour chacune des valeurs propres, déterminer l'espace propre associé.*

3. *La matrice* $\mathbb{D}$ *est-elle diagonalisable? Justifier.*

1

Exercice A.3.3

Q. 1 *Soit* $\mathbb{T} \in \mathcal{M}_{n,n}(\mathbb{C})$ *une matrice triangulaire supérieure. Montrer que si* $\mathbb{T}$ *est une matrice normale alors elle est diagonale.*

Q. 2 *Montrer que* $\mathbb{A} \in \mathcal{M}_{n,n}(\mathbb{C})$ *est une matrice normale si et seulement si il existe* $\mathbb{U} \in \mathcal{M}_{n,n}(\mathbb{C})$ *unitaire et* $\mathbb{D} \in \mathcal{M}_{n,n}(\mathbb{C})$ *diagonale telle que* $\mathbb{A} = \mathbb{U}\mathbb{D}\mathbb{U}^*$.

Q. 3 *En déduire qu'une matrice normale est diagonalisable et que ses vecteurs propres sont orthogonaux.*

2

Exercice A.3.4

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice hermitienne

Q. 1 *Montrer que*

$$\langle \mathbb{A}\boldsymbol{u}, \boldsymbol{u} \rangle \in \mathbb{R}, \ \forall \boldsymbol{u} \in \mathbb{C}^n. \tag{A.50}$$

On suppose de plus que la matrice $\mathbb{A}$ est définie positve.

Q. 2 1. *Montrer que les éléments diagonaux de* $\mathbb{A}$ *sont strictement positifs.*

2. *Montrer que les sous matrices principales de* $\mathbb{A}$ *sont elles aussi hermitiennes et définies positves.*

3

4

Exercice A.3.5: Procédé de Gram-Schmidt

rice:exo:19}

Soit $\{\boldsymbol{v}_i\}_{i\in[\![1,n]\!]}$ une base de $\mathbb{K}^n$. On construit successivement les vecteurs $\boldsymbol{u}_i$

$$\boldsymbol{u}_i = \boldsymbol{v}_i - \sum_{k=1}^{i-1} \frac{\langle \boldsymbol{u}_k, \boldsymbol{v}_i\rangle}{\langle \boldsymbol{u}_k, \boldsymbol{u}_k\rangle}\boldsymbol{u}_k,\ \forall i \in [\![1,n]\!].$$

Montrer qu'ils forment une **base orthogonale** de $\mathbb{K}^n$ et que $\text{Vect}\,(\boldsymbol{u}_1,\dots,\boldsymbol{u}_i) = \text{Vect}\,(\boldsymbol{v}_1,\dots,\boldsymbol{v}_i)$, $\forall i \in [\![1,n]\!]$.

Correction Exercice A.3.5 Montrons par récurrence sur i que

$$(\mathcal{H})_i : \text{Vect}\,(\boldsymbol{u}_1,\dots,\boldsymbol{u}_i) \text{ est une famille orthogonale et } \text{Vect}\,(\boldsymbol{u}_1,\dots,\boldsymbol{u}_i) = \text{Vect}\,(\boldsymbol{v}_1,\dots,\boldsymbol{v}_i)$$

Initialisation : Pour $i = 1$, on a $\boldsymbol{u}_1 = \boldsymbol{v}_1$ et $(\mathcal{H})_1$ est vérifiée.

Hérédité : Soit $i < n$. Supposons $(\mathcal{H})_i$ vérifiée. Montrons alors que $(\mathcal{H})_{i+1}$ est vraie.
On a

$$\boldsymbol{u}_{i+1} = \boldsymbol{v}_{i+1} - \sum_{k=1}^{i} \frac{\langle \boldsymbol{u}_k, \boldsymbol{v}_{i+1}\rangle}{\langle \boldsymbol{u}_k, \boldsymbol{u}_k\rangle}\boldsymbol{u}_k. \tag{A.51}$$ {RSL:RE:exo:19:c

- En effectuant le produit scalaire de (A.51) par $\boldsymbol{u}_j$ avec $j \in [\![1,i]\!]$ on obtient

$$\langle \boldsymbol{u}_j, \boldsymbol{u}_{i+1}\rangle = \langle \boldsymbol{u}_j, \boldsymbol{v}_{i+1}\rangle - \sum_{k=1}^{i} \frac{\langle \boldsymbol{u}_k, \boldsymbol{v}_{i+1}\rangle}{\langle \boldsymbol{u}_k, \boldsymbol{u}_k\rangle}\langle \boldsymbol{u}_j, \boldsymbol{u}_k\rangle.$$

  Par hypothèse de récurrence, la famille $\text{Vect}\,(\boldsymbol{u}_1,\dots,\boldsymbol{u}_i)$ est orthogonale, c'est à dire $\forall (r,s) \in [\![1,i]\!]^2$, $\langle \boldsymbol{u}_r, \boldsymbol{u}_s\rangle = 0$ si $r \neq s$ et $\boldsymbol{u}_r \neq 0$. On obtient donc

$$\langle \boldsymbol{u}_j, \boldsymbol{u}_{i+1}\rangle = \langle \boldsymbol{u}_j, \boldsymbol{v}_{i+1}\rangle - \frac{\langle \boldsymbol{u}_j, \boldsymbol{v}_{i+1}\rangle}{\langle \boldsymbol{u}_j, \boldsymbol{u}_j\rangle}\langle \boldsymbol{u}_j, \boldsymbol{u}_j\rangle = 0,\ \forall j \in [\![1,i]\!].$$

- On montre maintenant par l'absurde que $\boldsymbol{u}_{i+1} \neq 0$.
  Supposons $\boldsymbol{u}_{i+1} = 0$. Alors de (A.51), on obtient

$$\boldsymbol{v}_{i+1} = \sum_{k=1}^{i} \frac{\langle \boldsymbol{u}_k, \boldsymbol{v}_{i+1}\rangle}{\langle \boldsymbol{u}_k, \boldsymbol{u}_k\rangle}\boldsymbol{u}_k$$

  et donc $\boldsymbol{v}_{i+1} \in \text{Vect}\,(\boldsymbol{u}_1,\dots,\boldsymbol{u}_i) \overset{(\mathcal{H})_i}{=} \text{Vect}\,(\boldsymbol{v}_1,\dots,\boldsymbol{v}_i)$. Ceci entre en contradiction avec $\text{Vect}\,(\boldsymbol{v}_1,\dots,\boldsymbol{v}_n)$ base de $\mathbb{K}^n$.
- On déduit de (A.51) que $\boldsymbol{u}_{i+1} \in \text{Vect}\,(\boldsymbol{u}_1,\dots,\boldsymbol{u}_i,\boldsymbol{v}_{i+1})$. Par hypothèse de récurrence, $\text{Vect}\,(\boldsymbol{u}_1,\dots,\boldsymbol{u}_i) = \text{Vect}\,(\boldsymbol{v}_1,\dots,\boldsymbol{v}_i)$, ce qui donne $\boldsymbol{u}_{i+1} \in \text{Vect}\,(\boldsymbol{v}_1,\dots,\boldsymbol{v}_{i+1})$ et donc

$$\text{Vect}\,(\boldsymbol{u}_1,\dots,\boldsymbol{u}_{i+1}) = \text{Vect}\,(\boldsymbol{v}_1,\dots,\boldsymbol{v}_{i+1}).$$

◇

Exercice A.3.6: factorisation $\mathbb{QR}$

{RE:AL:matrice:exo:

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice inversible. Pour tout $i \in [\![1,n]\!]$, on note $\boldsymbol{a}_i = \mathbb{A}_{:,i}$ ses n vecteurs colonnes. En utilisant le procédé de Gram-schmidt sur la base $\{\boldsymbol{a}_1,\dots,\boldsymbol{a}_n\}$ montrer qu'il existe une matrice $\mathbb{Q}$ unitaire et une matrice triangulaire supérieure $\mathbb{R}$ à coefficients diagonaux strictement positifs tel que $\mathbb{A} = \mathbb{QR}$.

Correction Exercice A.3.6 On utilise le procédé d'orthonormalisation de Gram-Schmidt (voir Proposition A.12, page 114) pour obtenir la base orthogonale $\{\boldsymbol{u}_1,\dots,\boldsymbol{u}_n\}$ en calculant successivement

$$\boldsymbol{u}_i = \boldsymbol{a}_i - \sum_{k=1}^{i-1} \frac{\langle \boldsymbol{u}_k, \boldsymbol{a}_i\rangle}{\langle \boldsymbol{u}_k, \boldsymbol{u}_k\rangle}\boldsymbol{u}_k,\ \forall i \in [\![1,n]\!]. \tag{A.52}$$ {RSL:RE:exo:18:c

De plus on a $\text{Vect}\left(\boldsymbol{u}_1,\dots,\boldsymbol{u}_i\right)=\text{Vect}\left(\boldsymbol{a}_1,\dots,\boldsymbol{a}_i\right),\ \forall i\in[\![1,n]\!]$ On normalise la base orthogonale $\{\boldsymbol{u}_1,\dots,\boldsymbol{u}_n\}$ pour obtenir la base orthonormée $\mathcal{Q}=\{\boldsymbol{q}_1,\dots,\boldsymbol{q}_n\}$:

$$\boldsymbol{q}_i=\frac{\boldsymbol{u}_i}{\|\boldsymbol{u}_i\|_2},\ \forall i\in[\![1,n]\!]$$

et l'on a aussi $\text{Vect}\left(\boldsymbol{q}_1,\dots,\boldsymbol{q}_i\right)=\text{Vect}\left(\boldsymbol{a}_1,\dots,\boldsymbol{a}_i\right),\ \forall i\in[\![1,n]\!]$.
On note $\mathbb{Q}\in\mathcal{M}_n(\mathbb{K})$ la matrice définie par

$$\mathbb{Q}=\left(\begin{array}{c|c|c}&&\\ \boldsymbol{q}_1&\cdots&\boldsymbol{q}_n\\ &&\end{array}\right)$$

Cette matrice est clairement unitaire puisque la base $\{\boldsymbol{q}_1,\dots,\boldsymbol{q}_n\}$ est orthonormée.
Montrons que $\mathbb{Q}^*\mathbb{A}$ est triangulaire supérieure. On a

$$\mathbb{Q}^*\mathbb{A}=\left(\begin{array}{c}\boldsymbol{q}_1^*\\ \hline \vdots\\ \hline \boldsymbol{q}_n^*\end{array}\right)\left(\begin{array}{c|c|c}&&\\ \boldsymbol{a}_1&\cdots&\boldsymbol{a}_n\\ &&\end{array}\right)=\left(\begin{array}{c|c|c}\langle\boldsymbol{q}_1,\boldsymbol{a}_1\rangle&\cdots&\langle\boldsymbol{q}_1,\boldsymbol{a}_n\rangle\\ \hline \vdots&\ddots&\vdots\\ \hline \langle\boldsymbol{q}_n,\boldsymbol{a}_1\rangle&\cdots&\langle\boldsymbol{q}_n,\boldsymbol{a}_n\rangle\end{array}\right)$$

c'est à dire $(\mathbb{Q}^*\mathbb{A})_{i,j}=\langle\boldsymbol{q}_i,\boldsymbol{a}_j\rangle,\ \forall(i,j)\in[\![1,n]\!]$. Par définition cette matrice est triangulaire supérieure si $(\mathbb{Q}^*\mathbb{A})_{i,j}=0$ pour tout $i>j$. Soit $i\in[\![1,n-1]\!]$. La base $\mathcal{Q}$ étant orthonormée, on a $\boldsymbol{q}_i\perp\text{Vect}\left(\boldsymbol{q}_1,\dots,\boldsymbol{q}_{i-1}\right)$.
Comme $\text{Vect}\left(\boldsymbol{q}_1,\dots,\boldsymbol{q}_{i-1}\right)=\text{Vect}\left(\boldsymbol{a}_1,\dots,\boldsymbol{a}_{i-1}\right)$, on en déduit que

$$\langle\boldsymbol{q}_i,\boldsymbol{a}_j\rangle=0,\ \forall j\in[\![1,i-1]\!].$$

La matrice $\mathbb{Q}^*\mathbb{A}$ est donc triangulaire supérieure.
De plus, on a

$$(\mathbb{Q}^*\mathbb{A})_{i,i}=\langle\boldsymbol{q}_i,\boldsymbol{a}_i\rangle=\frac{\langle\boldsymbol{u}_i,\boldsymbol{a}_i\rangle}{\|\boldsymbol{u}_i\|_2}$$

En prenant le produit scalaire de (A.52) avec $\boldsymbol{u}_i$ on obtient

$$\begin{aligned}\langle\boldsymbol{u}_i,\boldsymbol{u}_i\rangle&=\langle\boldsymbol{u}_i,\boldsymbol{a}_i\rangle-\sum_{k=1}^{i-1}\frac{\langle\boldsymbol{u}_k,\boldsymbol{a}_i\rangle}{\langle\boldsymbol{u}_k,\boldsymbol{u}_k\rangle}\langle\boldsymbol{u}_i,\boldsymbol{u}_k\rangle\\&=\langle\boldsymbol{u}_i,\boldsymbol{a}_i\rangle\quad\text{car }\langle\boldsymbol{u}_i,\boldsymbol{u}_k\rangle=0,\ \forall k\neq i\text{ (base orthogonale)}\end{aligned}$$

Comme $\boldsymbol{u}_i\neq0$, on obtient $(\mathbb{Q}^*\mathbb{A})_{i,i}>0$.
On note $\mathbb{R}=\mathbb{Q}^*\mathbb{A}$ cette matrice triangulaire supérieure avec $R_{i,i}>0,\ \forall i\in[\![1,n]\!]$. La matrice $\mathbb{Q}$ étant unitaire (i.e. $\mathbb{Q}\mathbb{Q}^*=\mathbb{I}$) alors $\mathbb{A}=\mathbb{Q}\mathbb{R}$. ◇

Inverse d'une matrice

Exercice A.3.7

Soit $\mathbb{A}\in\mathcal{M}_3(\mathbb{R})$ définie par

$$\mathbb{A}=\begin{pmatrix}1&1&1\\1&2&2\\1&2&3\end{pmatrix}$$

Q. 1 *Calculer le déterminant de la matrice $\mathbb{A}$. Que peut-on en conclure?*

Q. 2 *Calculer si possible l'inverse de la matrice $\mathbb{A}$ en utilisant la technique de la matrice augmentée.*

Exercice A.3.8

Soient $\mathbb{A}$ et $\mathbb{B}$, deux matrices de $\mathcal{M}_n(\mathbb{K})$.

Q. 1 *Montrer que*

$$\mathbb{AB} = \mathbb{I} \Rightarrow \mathbb{BA} = \mathbb{I} \tag{A.53}$$

Conclure.

Exercice A.3.9

Q. 1 *Soit $\mathbb{A}$ une matrice inversible et symétrique, montrer que $\mathbb{A}^{-1}$ est symétrique.*

Q. 2 *Soit $\mathbb{A}$ une matrice carrée telle que $\mathbb{I} - \mathbb{A}$ est inversible. Montrer que*

$$\mathbb{A}\left(\mathbb{I} - \mathbb{A}\right)^{-1} = \left(\mathbb{I} - \mathbb{A}\right)^{-1}\mathbb{A}.$$

Q. 3 *Soient $\mathbb{A}$, $\mathbb{B}$ des matrices carrées inversibles de même dimension telle que $\mathbb{A}+\mathbb{B}$ soit inversible. Montrer que*

$$\mathbb{A}\left(\mathbb{A}+\mathbb{B}\right)^{-1}\mathbb{B} = \mathbb{B}\left(\mathbb{A}+\mathbb{B}\right)^{-1}\mathbb{A} = \left(\mathbb{A}^{-1}+\mathbb{B}^{-1}\right)^{-1}$$

Exercice A.3.10

{RE:AL:inverse:exo:

Soit $\mathbb{L} \in \mathcal{M}_n(\mathbb{C})$ une matrice triangulaire inférieure.

Q. 1 *A quelle(s) condition(s) la matrice $\mathbb{L}$ est-elle inversible?*

On suppose $\mathbb{L}$ inversible et on note $\mathbb{X} = \mathbb{L}^{-1}$.

Q. 2 *Montrer que $\mathbb{X}$ est une matrice triangulaire inférieure avec*

$$X_{i,i} = \frac{1}{L_{i,i}}, \ \forall i \in [\![1,n]\!].$$

Correction Exercice A.3.10

Q. 1 La matrice $\mathbb{L}$ est inversible si et seulement si son déterminant est non nul. Or le déterminant d'une matrice triangulaire est égal au produit de ses éléments diagonaux. Pour avoir $\mathbb{L}$, matrice triangulaire, inversible, il est nécessaire et suffisant d'avoir

$$L_{ii} \neq 0, \ \forall i \in [\![1,n]\!].$$

Q. 2 La matrice $\mathbb{X}$ étant la matrice inverse de $\mathbb{L}$, on a

$$\mathbb{LX} = \mathbb{I} \tag{A.54}$$

{RE:AL:Inverse:e

On note $\boldsymbol{X}^{[j]} = \mathbb{X}_{:,j}$ le j-ème vecteur colonne de la matrice $\mathbb{X}$ et $\boldsymbol{e}^{[j]}$ le j-ème vecteur de la base canonique de $\mathbb{C}^n$ ($e_i^{[j]} = \delta_{i,j}$).
L'équation (A.54) peut donc se réécrire

$$\mathbb{L}\left(\begin{array}{c|c|c} & & \\ \boldsymbol{X}^{[1]} & \cdots & \boldsymbol{X}^{[n]} \\ & & \end{array}\right) = \left(\begin{array}{c|c|c} & & \\ \boldsymbol{e}^{[1]} & \cdots & \boldsymbol{e}^{[n]} \\ & & \end{array}\right)$$

ou encore, déterminer la matrice $\mathbb{X}$ inverse de $\mathbb{L}$ reviens à résoudre les n systèmes linéaires suivants:

$$\mathbb{L}\boldsymbol{X}^{[j]} = \boldsymbol{e}^{[j]}, \quad \forall j \in [\![1,n]\!]. \tag{A.55}$$

{RE:AL:inverse:e

- Pour montrer que $\mathbb{X}$ est triangulaire inférieure il suffit de vérifier que pour tout $j \in [\![2,n]\!]$

$$X_i^{[j]} = 0, \quad \forall i \in [\![1,j-1]\!].$$

Soit $j \in [\![2,n]\!]$. On décompose la matrice $\mathbb{L}$ en la matrice bloc carré 2 par 2 ou le premier bloc diagonal, noté $\mathbb{L}_{j-1}$, est une matrice triangulaire inférieure inversible de dimension $j-1$. Le système (A.55) s'écrit alors

$$\mathrm{L}\mathbf{X}^{[j]} = \boldsymbol{e}^{[j]} \iff \left(\begin{array}{ccc:c} L_{1,1} & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ \bullet & \cdots & L_{j-1,j-1} & 0 \\ \hdashline \bullet & \cdots & \bullet & L_{j,j} \end{array}\right) \left(\begin{array}{c} X_1^{[j]} \\ \vdots \\ X_{j-1}^{[j]} \\ \hdashline X_j^{[j]} \\ \vdots \\ X_n^{[j]} \end{array}\right) = \left(\begin{array}{c} 0 \\ \vdots \\ 0 \\ \hdashline 1 \\ 0 \\ \vdots \\ 0 \end{array}\right) \quad (j-1)$$

On en déduit donc que nécessairement

$$\begin{pmatrix} L_{1,1} & 0 & \cdots & 0 \\ \bullet & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \bullet & \cdots & \bullet & L_{j-1,j-1} \end{pmatrix} \begin{pmatrix} X_1^{[j]} \\ \vdots \\ X_{j-1}^{[j]} \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$$

Comme la matrice de ce système est inversible, on a bien $X_i^{[j]} = 0$, $\forall i \in [\![1, j-1]\!]$ et donc la matrice $\mathbb{X}$ est triangulaire inférieure.

- Par définition du produit matricielle, de l'équation (A.54) on tire pour tout $j \in [\![1, n]\!]$

$$(\mathbb{L}\mathbb{X})_{j,j} = (\mathbb{I})_{j,j} \iff \sum_{k=1}^{n} L_{j,k} X_{k,j} = 1 \iff \sum_{k=1}^{j-1} L_{j,k} X_{k,j} + L_{j,j} X_{j,j} + \sum_{k=j+1}^{n} L_{j,k} X_{k,j} = 1$$

Or les matrices $\mathbb{L}$ et $\mathbb{X}$ sont triangulaires inférieures et donc $X_{k,j} = 0$, $\forall k \in [\![1, j-1]\!]$, et $L_{j,k} = 0$, $\forall k \in [\![j+1, n]\!]$. On obtient alors $L_{j,j} X_{j,j} = 1$ et comme $L_{j,j} \neq 0$ on a bien $X_{j,j} = 1/L_{j,j}$.

◇

Exercice A.3.11

Soit $\mathbb{A} \in \mathcal{M}_{n,n}(\mathbb{K})$ et $\mathbb{U}$, $\mathbb{B}$, $\mathbb{V}$ trois matrices rectangulaires.

Q. 1 *Sous quelles hypothèses peut-on définir la matrice* $\mathbb{G}$ *suivante*

$$\mathbb{G} = \mathbb{A}^{-1} - \mathbb{A}^{-1}\mathbb{U}\left(\mathbb{I} + \mathbb{B}\mathbb{V}\mathbb{A}^{-1}\mathbb{U}\right)^{-1}\mathbb{B}\mathbb{V}\mathbb{A}^{-1} ? \tag{A.56}$$

Q. 2 *Montrer que* $(\mathbb{A} + \mathbb{U}\mathbb{B}\mathbb{V})\,\mathbb{G} = \mathbb{I}$. *Conclure.*

Q. 3 *Soit* $\beta \in \mathbb{R}$ *et* $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{C}^n$. *Calculer* $(\mathbb{A} + \beta \boldsymbol{u}\boldsymbol{v}^t)^{-1}$.

Exercice A.3.12

Etant donnée une matrice $\mathbb{D} \in \mathcal{M}_{n,n}(\mathbb{C})$, on pose

$$\mathbb{D} = \mathbb{A} + \imath\mathbb{B} \text{ avec } \mathbb{A}, \mathbb{B} \in \mathcal{M}_{n,n}(\mathbb{R})$$

Sous certaines hypothèses à préciser, établir la relation

$$\mathbb{D}^{-1} = \left(\mathbb{A} + \mathbb{B}\mathbb{A}^{-1}\mathbb{B}\right)^{-1} - \imath\mathbb{A}^{-1}\mathbb{B}\left(\mathbb{A} + \mathbb{B}\mathbb{A}^{-1}\mathbb{B}\right)^{-1}.$$

Matrices blocs

Exercice A.3.13

On considère les matrices blocs suivantes

$$\mathbb{A} = \left(\begin{array}{cc:cc} 1 & 2 & 1 & 0 \\ 3 & 4 & 0 & 1 \\ \hdashline 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{array}\right) = \left(\begin{array}{c:c} \mathbb{C} & \mathbb{I} \\ \hdashline \mathbb{I} & 0 \end{array}\right) \quad \text{et} \quad \mathbb{B} = \left(\begin{array}{cc:cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hdashline 1 & 2 & 1 & 2 \\ 3 & 4 & 3 & 4 \end{array}\right) = \left(\begin{array}{c:c} \mathbb{I} & 0 \\ \hdashline \mathbb{C} & \mathbb{C} \end{array}\right)$$

avec par identification

$$\mathbb{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{et} \quad \mathbb{C} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

Q. 1 *Calculer les matrices $\mathbb{AB}$ et $\mathbb{BA}$ en utilisant l'écriture bloc.*

Q. 2 *Exprimer les matrices $\mathbb{A}(\mathbb{A}+\mathbb{B})$ et $(2\mathbb{B}-\mathbb{A})(\mathbb{B}+\mathbb{A})$ en fonction des matrices $\mathbb{C}$ et $\mathbb{I}$.*

1

Exercice A.3.14

Soient $\mathbb{A} \in \mathcal{M}_{n,k}(\mathbb{K})$ et $\mathbb{B} \in \mathcal{M}_{k,n}(\mathbb{K})$. On note $\mathbb{L}$ la matrice

$$\mathbb{L} = \left(\begin{array}{c:c} \mathbb{I} - \mathbb{BA} & \mathbb{B} \\ \hdashline 2\mathbb{A} - \mathbb{ABA} & \mathbb{AB} - \mathbb{I} \end{array}\right).$$

Q. 1 *Montrer que la matrice $\mathbb{L}$ est bien définie et spécifier les dimensions des blocs.*

Q. 2 *Calculer $\mathbb{L}^2$. Que peut-on en conclure?*

2

Exercice A.3.15: *résultats à savoir*

Soient $\mathbb{A} \in \mathcal{M}_m(\mathbb{C})$, $\mathbb{B} \in \mathcal{M}_n(\mathbb{C})$ et $\mathbb{D} \in \mathcal{M}_{m,n}(\mathbb{C})$.

Q. 1 *Calculer, en fonction des déterminant de $\mathbb{A}$ et $\mathbb{B}$, le déterminant des matrices*

$$\mathbb{E} = \left(\begin{array}{c:c} \mathbb{A} & \mathbb{O} \\ \hdashline \mathbb{O} & \mathbb{I}_n \end{array}\right), \ \mathbb{F} = \left(\begin{array}{c:c} \mathbb{I}_m & \mathbb{O} \\ \hdashline \mathbb{O} & \mathbb{B} \end{array}\right), \quad \textit{et } \mathbb{G} = \left(\begin{array}{c:c} \mathbb{A} & 0 \\ \hdashline 0 & \mathbb{B} \end{array}\right).$$

Q. 2 *Soit $\mathbb{H} = \left(\begin{array}{c:c} \mathbb{A} & \mathbb{D} \\ \hdashline 0 & \mathbb{B} \end{array}\right)$. En utilisant les factorisations $\mathbb{QR}$ des matrices $\mathbb{A}$ et $\mathbb{B}$, montrer que*

$$\det(\mathbb{H}) = \det(\mathbb{A}) \det(\mathbb{B}). \tag{A.57}$$

Q. 3 *En déduire qu'**une matrice triangulaire supérieure par blocs est inversible si et seulement si ses matrices blocs diagonales sont inversibles**.*

Q. 4 *En déduire qu'**une matrice triangulaire inférieure par blocs est inversible si et seulement si ses matrices blocs diagonales sont inversibles**.*

3

A.4 Listings

A.4.1 Codes sur la méthode de dichotomie/bisection

{Sec:Annex

Transcription de l'Algorithme 3.1(page 28)

Listing A.1: fonction `dichotomie1` (Matlab) Code:Matlab:Dichotomie1

```
function x=dichotomie1(f,a,b,epsilon)
  kmin=floor( log((b-a)/epsilon)/log(2) );
  X=zeros(kmin+1,1);
  A=zeros(kmin+1,1);B=zeros(kmin+1,1);
  A(1)=a;B(1)=b;X(1)=(a+b)/2;
  for k=1:kmin
    if f(X(k))==0
      A(k+1)=X(k);B(k+1)=X(k);
    elseif f(B(k))*f(X(k))<0
      A(k+1)=X(k);B(k+1)=B(k);
    else
      A(k+1)=A(k);B(k+1)=X(k);
    end
    X(k+1)=(A(k+1)+B(k+1))/2;
  end
  x=X(kmin+1);
end
```

Listing A.2: fonction `dichotomie1` (C) Code:C:Dichotomie1

```
double dichotomie1(double (*f)(double),
        double a, double b,double eps){
  double *A,*B,*X,x;
  int kmin;
  size_t Size;
  assert(f(a)*f(b)<0);
  kmin=(int)floor(log((b-a)/eps)/log(2.));
  Size=(kmin+1)*sizeof(double);
  assert( X=(double*)malloc(Size) );
  assert( A=(double*)malloc(Size) );
  assert( B=(double*)malloc(Size) );
  // ou assert( A && B && X );
  A[0]=a;B[0]=b;X[0]=(a+b)/2.;
  for (k=0;k<kmin;k++){
    if (f(X[k])==0){
      A[k+1]=X[k];B[k+1]=X[k];
    }else if ( f(B[k])*f(X[k])<0 ){
      A[k+1]=X[k];B[k+1]=B[k];
    }else{
      A[k+1]=A[k];B[k+1]=X[k];
    }
    X[k+1]=(A[k+1]+B[k+1])/2;
  }
  x=X[kmin];
  free(X);free(A);free(B);
  return x;
}
```

Listing A.3: script `dichotomie1` (Matlab) Code:Matlab:mainDichotomie1

```
clear all
close all
f=@(x) (x+2)*(x+2)*(x-pi);
x=dichotomie1(f,-1,2*pi,1e-8);
fprintf('x=%.16f\n',x)
x=dichotomie1(@cos,2,pi,1e-8);
fprintf('x=%.16f\n',x)
```

Listing A.4: main `dichotomie1` (C) Code:C:mainDichotomie1

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <assert.h>

double dichotomie1(
  double (*f)(double),
  double a, double b, double eps
);
double g1(double x){
  return (x+2)*(x+2)*(x-M_PI);
}

int main(){
  double x;
  x=dichotomie1(g1,-1,2*M_PI,1e-8);
  printf("x=%.16lf, error=%.6e\n",
         x,fabs(x-M_PI));
  x=dichotomie1(cos,-1,M_PI,1e-8);
  printf("x=%.16lf, error=%.6e\n",
         x,fabs(x-M_PI_2));
  return 1;
}
// Definition de dichotomie1 ensuite ...
```

Transcription de l'Algorithme 3.5 (page 30)

Listing A.5: fonction `dichotomie5` (Matlab) Code:Matlab:Dichotomie5

```
function x=dichotomie5(f,a,b)
  assert(f(a)*f(b)<0, ...
         'test f(a)*f(b)<0 failed');
  A=a;B=b;x=(A+B)/2;xp=A;
  while x~=xp
    if f(B)*f(x)<0
      A=x;
    else
      B=x;
    end
    xp=x;
    x=(A+B)/2;
  end
end
```

Listing A.6: fonction `dichotomie5` (C) Code:C:Dichotomie5

```
double dichotomie5(double (*f)(double),
                   double a, double b){
  double A,B,x,xp;
  assert(f(a)*f(b)<0);
  A=a;B=b;x=(A+B)/2.;xp=A;
  while (x!=xp){
    if (f(B)*f(x)<0)
      A=x;
    else
      B=x;
    xp=x;
    x=(A+B)/2;
  }
  return x;
}
```

Listing A.7: script `dichotomie5` (Matlab) Code:Matlab:mainDichotomie5

```
clear all
close all
f=@(x) (x+2)*(x+2)*(x-pi);
x=dichotomie5(f,-1,2*pi);
fprintf('x=%.16f\n',x)
x=dichotomie5(@cos,2,pi);
fprintf('x=%.16f\n',x)
```

Listing A.8: main `dichotomie5` (C) Code:C:mainDichotomie5

```
#include <stdio.h>
#include <math.h>
#include <assert.h>

double dichotomie5(double (*f)(double),
                   double a, double b);
double g1(double x){
  return (x+2)*(x+2)*(x-M_PI);
}

int main(){
  double x;
  x=dichotomie5(g1,-1,2*M_PI);
  printf("x=%.16lf\n",x);
  x=dichotomie5(cos,-1,M_PI);
  printf("x=%.16lf\n",x);
```

Liste des algorithmes

Bibliography

[1] J.P. Demailly. *Analyse Numérique et Equations Différentielles*. PUG, 1994.

[2] W. Gander, M.J. Gander, and F. Kwok. *Scientific computing : an introduction using Maple and MATLAB*. Springer, Cham, 2014.

[3] Huckle T. Collection of software bugs.