Exercice 1

1. Qu'appelle-t-on « base canonique de $\mathbb{R}^3$ » ?
2. Montrer que les vecteurs (u, v, w) définis ci-dessous forment une base de $\mathbb{R}^3$:

$$u = (1, 2, 1), \qquad v = (2, 0, 1), \qquad w = (2, 1, 1).$$

On introduit l'endomorphisme f de $\mathbb{R}^3$ vérifiant

$$\begin{cases} f(u) &= u \\ f(v) &= u + v \\ f(w) &= w \end{cases}$$

3. Donner la matrice de f dans la base (u, v, w).
4. Donner la matrice de f dans la base canonique.

Exercice 2

On définit la matrice $Q = \begin{pmatrix} 1 & 5 & -2 \\ 0 & -1 & 1 \\ 1 & 5 & -1 \end{pmatrix}$.

1. Montrer que Q est inversible et calculer son inverse.

Soient

- E : $\mathbb{C}$-espace vectoriel de dimension 2, muni d'une base : $b_E = (i, j)$;
- F : $\mathbb{C}$-espace vectoriel de dimension 3, muni d'une base : $b_F = (u, v, w)$;
- φ : morphisme de $E \to F$ dont la matrice entre b_E et b_F est

$$M = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

2. (a) Déterminer $\mathrm{Im}(f)$ et $\mathrm{Ker}(f)$.
   (b) f est-il injectif ? surjectif ?
3. Montrer que la famille $B_F = (4u - v - w, 5u - v, -3u + v + w)$ est une autre base de F.
4. Exprimer $f(2i - j)$ dans B_F.

   Montrer que l'on retrouve ce résultat en calculant $Q \times M \times \begin{pmatrix} 2 \\ -1 \end{pmatrix}$.

Exercice 3

Pour un ensemble E et $f : E \longrightarrow E$, on définit par récurrence sur $k \in \mathbb{N}$ l'application $f^k : E \longrightarrow E$ de la façon suivante :

$$f^0 = \mathrm{Id}_E \qquad \text{et} \qquad f^{k+1} = f \circ f^k.$$

Si f est bijective, on note sa réciproque f^{-1} et pour tout $k \in \mathbb{N}$, on note

$$f^{-k} = \left(f^{-1}\right)^k.$$

Soit un entier strictement positif : n.

A. L'opérateur de translation est l'endomorphisme τ de $\mathbb{R}_n[X]$ défini par

$$\tau \;:\; \begin{array}{rcl} \mathbb{R}_n[X] & \longrightarrow & \mathbb{R}_n[X] \\ P(X) & \mapsto & P(X+1). \end{array}$$

1. Pour tout polynôme non-nul $P \in \mathbb{R}_n[X]$, exprimer le degré et le coefficient dominant de $\tau(P)$ en fonction de ceux P.
2. Soit $P \in \mathbb{R}_n[X]$. Pour tout $k \in \mathbb{N}$, donner une expression de $\tau^k(P)$ en fonction de P.
3. Donner la matrice $M = (m_{i,j})_{(i,j)\in[\![1,n]\!]^2}$ de l'endomorphisme τ dans la base canonique de $\mathbb{R}_n[X]$. On exprimera les coefficients en fonction de i et j.
4. L'application τ est-elle bijective ? Si oui, préciser τ^{-1}. L'expression trouvée au A.2 est-elle aussi valable pour $k \in \mathbb{Z}$?
5. Que vaut $M^{-1} = \left(m'_{i,j}\right)_{(i,j)\in[\![1,n]\!]^2}$? On exprimera les coefficients en fonction de i et j.

B. L'opérateur de différence est l'endomorphisme δ de $\mathbb{R}_n[X]$ défini par

$$\begin{array}{rccc} \delta \; : & \mathbb{R}_n[X] & \longrightarrow & \mathbb{R}_n[X] \\ & P(X) & \mapsto & P(X+1). \end{array} \qquad \text{on a donc : } \delta = \tau - \mathrm{Id}_{\mathbb{R}_n[X]}$$

1. Pour tout polynôme non-constant $P \in \mathbb{R}_n[X]$, exprimer le degré et le coefficient dominant de $\delta(P)$ en fonction de P.
2. En utilisant celle de τ, donner la matrice $N = (n_{i,j})_{(i,j)\in[\![1,n]\!]^2}$ de l'endomorphisme δ dans la base canonique de $\mathbb{R}_n[X]$. On exprimera les coefficients en fonction de i et j.
3. En déduire le noyau et l'image de l'endomorphisme δ.

C. Changement de bases. On définit la famille de polynômes $(B_k)_{k\in[\![0,n]\!]}$ par $B_0 = 1$ et

$$\forall k \in [\![1,n]\!],\; B_k(X) = \frac{X(X-1)\cdots(X-k+1)}{k!} = \frac{1}{k!}\prod_{i=0}^{k-1}(X-i).$$

1. Montrer que cette famille est une base de $\mathbb{R}_n[X]$.
2. Établir que, pour tout couple d'entiers naturels (i,k), $B_k(i)$ et $B_k(-i)$ sont des entiers.
3. Calculer $\delta(B_0)$, puis $\delta(B_{k+1})$ en fonction de B_k pour $k \in [\![1,n]\!]$.
4. Donner la matrice $N' = \left(n'_{i,j}\right)_{(i,j)\in[\![1,n]\!]^2}$ de l'endomorphisme δ dans cette nouvelle base.

D. Application à l'interpolation

Soit $\phi : \left\{ \begin{array}{ccc} \mathbb{R} & \longrightarrow & \mathbb{R} \\ t & \mapsto & \phi(t), \end{array} \right.$ une fonction. L'opérateur δ qui était jusqu'ici réservé aux polynômes est maintenant étendu aux fonctions : $\delta\phi$ représente la fonction $x \mapsto \phi(x+1) - \phi(x)$.

Soient t_0, $t_1 = t_0 + h$, $t_2 = t_0 + 2h$, $\ldots$, $t_n = t_0 + nh$: $n+1$ points régulièrement espacés ($h > 0$). On dit que le polynôme Q **interpole** ϕ aux points $t_0, t_1, \ldots, t_n$ si $\forall i \in [\![0,n]\!]$, $Q(t_i) = \phi(t_i)$.
Le changement de variable $x = \frac{t-t_0}{h}$ est tel que $t = t_i$ si et seulement si $x = i$. On voit donc que

$$\Big(Q(t_i) = \phi(t_i)\Big) \iff \Big(P(i) = f(i)\Big)$$

où l'on a posé $P(x) = Q(t) = Q(t_0 + xh)$ et $f(x) = \phi(t) = \phi(t_0 + xh)$.

Dans la suite de ce problème, on cherche une méthode efficace pour trouver (et évaluer) le polynôme P de $\mathbb{R}_n[X]$ interpolant f aux points $0, 1, \ldots, n$.

L'idée majeure est d'exprimer P dans la base $(B_k)_{k\in[\![0,n]\!]}$.

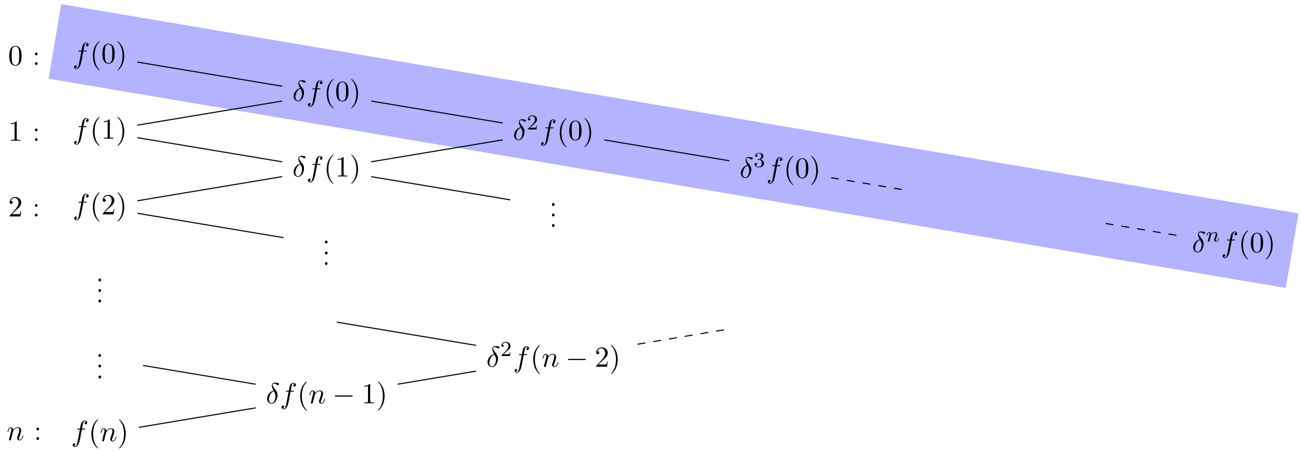
1. Montrer par récurrence que $\forall j \in \mathbb{N},\; (\delta^j f)(x) = \displaystyle\sum_{i=0}^{j}(-1)^{j-i}\binom{j}{i} f(x+i)$.
2. En déduire que la condition $\forall i \in [\![0,n]\!],\; P(i) = f(i)$ implique $\forall j \in [\![0,n]\!],\; (\delta^j P)(0) = (\delta^j f)(0)$.
3. Pour un k fixé, que vaut $(\delta^j B_k)(0)$? En déduire, si $P = \displaystyle\sum_{k=0}^{n}\lambda_k B_k$, une autre expression de $(\delta^j P)(0)$.

Ceci permet de conclure avec la (jolie) formule :

> Le polynôme de degré $\leqslant n$ qui interpole f aux points $0, 1, \ldots, n$ s'écrit
>
> $$P = \sum_{k=0}^{n} \frac{(\delta^k f)(0)}{k!} X(X-1)\cdots(X-k+1).$$

... formule due à Newton, qui nous rappelle vaguement celle de Taylor.

La présentation suivante, très pratique, donne les coefficients $(\delta^k f)(0)$:



Maintenant, si on veut évaluer le polynôme P efficacement (par exemple pour placer beaucoup de points de sa courbe représentative à l'écran d'un ordinateur), on peut profiter de l'égalité suivante :

$$\sum_{k=0}^{n} \lambda_k \frac{X(X-1)\cdots(X-k+1)}{k!} = \lambda_0 + X\left(\lambda_1 + \frac{X-1}{2}\Big(\lambda_2 + \frac{X-2}{3}\big(\cdots + \frac{X-n+1}{n}(\lambda_n)\cdots\big)\Big)\right)$$

Ainsi, l'algorithme suivant permet, en un nombre record de calculs, de donner $P(x)$ connaissant x et la liste `L`$= (\lambda_0, \ldots, \lambda_n)$:

```
def P(L,x):
  """ Evluates y= P(L,x) at x where P is a polynomial whose coefficients
      in the basis ( 1, X, X(X-1)/2, ..., X(X-1)...(X-n+1)/n! ) are
      listed in L.
      NB : degree(P) <= n = len(L) - 1
  """
  n = len(L) - 1
  y = 0.0
  for i in range(n,-1,-1):
    y = y * (x-i) / (i+1.) + L[i]
  return y
```

… algorithme qui nous rappelle vaguement celui de Ruffini.

4. (Exemple) Donner le polynôme qui interpole la fonction $x \mapsto \sin\left(\frac{\pi x}{2}\right)$ aux points $\{0, 1, 2, 3, 4\}$. Vérifier que votre polynôme possède bien 0, 2 et 4 comme racines.
5. Écrire l'algorithme qui donne la liste `L` à partir de la liste des valeurs `[`$f(0), f(1), \ldots, f(n)$`]`.