Programme de colle 25 - Semaine du 28 avril 2025

Informatique (voir exemples en annexe)

- **Tableaux 2D** : commande `import numpy as np` et utilisation de `array`, `linspace`, `zeros`, `shape` etc et manipulations de base pour parcourir les éléments d'un tableau 2D, commande `M[i,j]`, etc..

Chapitre 19 : Limites et comparaisons de fonctions

- Définitions avec les quantificateurs des limites suivantes :
  $\lim\limits_{x\to x_0} f(x)=\ell$, $\lim\limits_{x\to x_0} f(x)=+\infty$, $\lim\limits_{x\to x_0} f(x)=-\infty$, $\lim\limits_{x\to+\infty} f(x)=\ell$, $\lim\limits_{x\to+\infty} f(x)=+\infty$, ...
- Limite à gauche et limite à droite en un point, définition avec les quantificateurs.
- Asymptote verticale et horizontale.
- Opérations sur les limites : addition, multiplication, quotient, composition, formes indéterminées.
- Limites et inégalités : Si $\lim\limits_{x\to x_0} f(x)=l>0$ alors il existe un voisinage de x_0 sur lequel f est strictement positive, théorème des gendarmes, théorèmes de comparaison.
- Une fonction monotone sur un intervalle ouvert admet une limite en chacune des bornes de l'intervalle.
- Croissances comparées (logarithme, puissance, exponentielle)
- Fonctions équivalentes : définition et équivalents usuels :
  - polynômes en $+\infty$, $-\infty$ et 0.
  - $\sin(x) \underset{x\to0}{\sim} x$
  - $\tan(x) \underset{x\to0}{\sim} x$
  - $\ln(1+x) \underset{x\to0}{\sim} x$
  - $e^x-1 \underset{x\to0}{\sim} x$
  - pour tout $\alpha\in\mathbb{R}^*$, $(1+x)^\alpha-1 \underset{x\to0}{\sim} \alpha x$ et en particulier (pour $\alpha=\frac{1}{2}$), $\sqrt{1+x}-1 \underset{x\to0}{\sim} \frac{x}{2}$
  - $1-\cos(x) \underset{x\to0}{\sim} \frac{x^2}{2}$
- Propriétés des équivalents : transitivité, produit, quotient, puissance (pas de somme ni de composition !), théorème de substitution.

Chapitre 20 : Statistiques univariées

- Définitions : Série statistiques, médiane, quartiles, écart interquartiles, moyenne (linéarité de la moyenne Démonstration exigible).
- Variance : définition, théorème de Koenig-huygens (Démonstration exigible), formule $V(ax_i+b)=a^2V(x_i)$ (Démonstration exigible)
- Ecart-type.

Chapitre 21 : Applications linéaires

- Applications linéaires
  - Définition d'une application linéaire, image du vecteur nul, vocabulaire (endomorphisme, isomorphisme, automorphisme)
  - Opérations : somme, multiplication par un scalaire, composition (démonstration exigible), réciproque d'applications linéaires.

Remarques aux colleurs

1. Merci de commencer par poser une question de cours, puis une question d'informatique.
2. **Attention au décalage entre les programmes des classe de BCPST1A et BCPST1B**. Le chapitre statistique n'a pas été traité en 1A, nous sommes allés moins loin sur les applications linéaires en 1B.

Annexe 1 : Exemples

Exemples de question d'informatique

1. Ecrire en Python une fonction `existence` qui prend en entrée un tableau 1D `T` et un nombre `element` et renvoie `True` si `element` se trouve dans le tableau `T`, `False` sinon.

```
def existence(T,element):
    a=len(T)                # nombre de lignes
    for i in range(a):      # parcours des lignes
        if T[i]==element:   # on teste si T[i] est egal a element
            return True
    return False            # si element n est pas dans T
```

2. Ecrire en Python une fonction `MaximumTableau` qui prend en entrée un tableau 1D `T` et renvoie la plus grande valeur de ce tableau.

```
def MaximumTableau(T):
    a=len(T)                # nombre de lignes
    maxi = T[0]             # initialisation avec la 1ere valeur du tableau
    for i in range(a):      # parcours des lignes
        if T[i]>maxi:       # on teste si T[i] est plus grand
            maxi=T[i]       # on a trouve une plus grande valeur
    return maxi
```

3. Ecrire en Python une fonction `Moyenne` qui prend en entrée un tableau 1D `T` et renvoie la moyenne de ses éléments :

```
def Moyenne(T):
    a=len(T)                # nombre de lignes
    S=0                     # initialisation de la somme
    for i in range(a):      # parcours des lignes
        S=S+T[i]            # on ajoute T[i] a la somme
    return S/a              # formule pour la moyenne
```