**Syllabus du Master en Intelligence Artificielle et Modélisation de Données**

**SEMESTRE 7**

|  |
| --- |
|  |

**Outils Mathématiques de données**

**Volume horaire :**

* **Cours magistral (CM)** : 24h
* **Travaux dirigés (TD)** : 15h

**Pré-requis :**  
Notions de base en algèbre linéaire (vecteurs, matrices, diagonalisation), en calcul différentiel (dérivées partielles) et en analyse multivariée.

### ****Description :****

Ce cours introduit les concepts mathématiques fondamentaux pour aborder des sujets avancés en modélisation, optimisation et intelligence artificielle. L’accent est mis sur l'algèbre linéaire appliquée à des modèles de données, ainsi que sur l'analyse multivariée nécessaire pour l’optimisation et la modélisation scientifique.

### ****Objectifs :****

* Acquérir une compréhension approfondie de l’algèbre linéaire appliquée aux modèles de données.
* Appréhender les outils d’analyse multivariée utilisés en optimisation et modélisation.
* Appliquer les concepts théoriques à des problèmes pratiques, notamment dans le cadre de la résolution d’optimisation et d’analyse de données.
* Préparer les étudiants à des cours avancés en modélisation mathématique, optimisation et IA.

### ****Contenu détaillé :****

#### **1. Algèbre linéaire appliquée à la modélisation**

* **Réduction des endomorphismes**
  + **Valeurs propres et vecteurs propres** : étude des matrices carrées, diagonalisation, impact sur la réduction de dimension.
  + **Changements de base et trigonalisation** : simplification de la structure matricielle et transformation des bases.
  + **Théorème de Cayley-Hamilton** : relation entre la matrice et son polynôme caractéristique.
  + **Critères de diagonalisabilité** : conditions sous lesquelles une matrice est diagonalisable et exemples concrets (projecteurs, symétries).
* **Espaces euclidiens et formes quadratiques**
  + **Formes quadratiques** : définition et propriétés, identification des formes positives et négatives.
  + **Vecteurs isotropes** : propriétés des vecteurs qui restent invariants sous une transformation linéaire.
  + **Produit scalaire et norme associée** : lien avec l’orthogonalité et la projection des vecteurs.
  + **Bases orthonormées** : méthode de Gram-Schmidt et formule de Parseval pour la décomposition des vecteurs.
  + **Inégalité de Cauchy-Schwarz** : application en optimisation et en modélisation.
* **Endomorphismes symétriques et décompositions matricielles**
  + **Adjoint d’un endomorphisme** : définition et applications.
  + **Diagonalisation en base orthonormée** : simplification et calcul des valeurs propres pour des matrices symétriques.
  + **Décompositions $LD^t L$, $P^t P$** : analyse des propriétés spectrales.
  + **Applications pratiques** : matrices de variance-covariance, matrices d’inertie, et leur décomposition via SVD.

#### **2. Analyse multivariée pour l’optimisation et la modélisation**

* **Calcul différentiel et dérivées partielles**
  + **Gradient et Jacobienne** : notions de base pour les fonctions multivariées, et leur utilisation en optimisation (descente de gradient).
  + **Différentiabilité et théorème de Fubini** : conditions de continuité et changement de variables dans des fonctions multivariées.
* **Dérivées d'ordre 2 et analyse des comportements locaux**
  + **Dérivées secondes et matrice Hessienne** : critère d'extrémalité pour une fonction multivariée.
  + **Théorème de Schwarz** : symétrie des dérivées secondes et implication pour la modélisation.
  + **Formule de Taylor d'ordre 2** : approximation des fonctions multivariées au voisinage d'un point.
* **Étude des extrema**
  + **Conditions nécessaires d'extrémalité** : critères pour déterminer les points stationnaires et les points extrêmes d'une fonction.
  + **Critères de Monge** : application dans les fonctions à plusieurs variables pour identifier les minima et maxima.
  + **Méthode des multiplicateurs de Lagrange** : approche pour les problèmes d'optimisation avec contraintes, et applications concrètes.

### ****Compétences visées :****

* Utiliser les outils d’algèbre linéaire pour résoudre des systèmes complexes de données et simplifier des modèles.
* Appliquer les concepts d’analyse multivariée pour étudier des fonctions et résoudre des problèmes d'optimisation.
* Maîtriser les techniques d’analyse des extrema et de résolution de problèmes avec contraintes (Lagrange).
* Développer une compréhension mathématique de la modélisation des données et des systèmes complexes.

**Programmation algorithmique et Base de données**

**Volume horaire :**

* **Cours Magistral (CM)** : 25,5h
* **Travaux Dirigés (TD)** : 21h
* **Travaux Pratiques (TP)** : 16h

**Pré-requis :**  
Notions de base en algorithmique, mathématiques discrètes et statistiques. Une connaissance élémentaire du langage Python est un plus.

### ****Description générale :****

Ce cours couvre l'intégration de la programmation Python avec des concepts clés de la gestion des bases de données, de l'algorithmique et de l'analyse de données. Il inclut des techniques avancées dans la manipulation de données, la modélisation de bases de données relationnelles et la création de visualisations interactives. L'accent est mis sur la programmation orientée objet, l'algorithmique, ainsi que l'utilisation d'outils de data science et de business intelligence (SQL, Power BI, PyTorch, OpenCV).

### ****Objectifs :****

* **Maîtriser Python** et ses bibliothèques pour le traitement et l'analyse des données (NumPy, Pandas).
* **S'initier à la programmation orientée objet (POO)** et comprendre les principes fondamentaux de l'algorithmique (récursivité, complexité, structures de données).
* **Concevoir et manipuler des bases de données relationnelles** (modélisation, création et gestion de données).
* **Maîtriser SQL** pour interroger et manipuler des bases de données.
* **Créer des visualisations interactives** et des tableaux de bord avec Power BI.
* **Découvrir les bases de l'apprentissage profond** (deep learning) et le traitement d'image avec **PyTorch** et **OpenCV**.

### ****Plan du cours détaillé :****

#### **1. Variables, fonctions et structures de contrôle en Python**

* **Variables** : types primitifs (entiers, flottants, chaînes de caractères, booléens).
* **Opérations sur les variables** : opérateurs arithmétiques, logiques, relations et affectation.
* **Structures de contrôle** : conditions (if, else, elif), boucles (for, while), gestion des exceptions.
* **Fonctions** : déclaration, arguments, retour de valeurs, portée des variables, récursivité.

#### **2. Programmation orientée objet (POO)**

* **Principes de la POO** : classes, objets, attributs, méthodes.
* **Encapsulation, héritage et polymorphisme** : concepts clés de la POO appliqués à la programmation Python.
* **Gestion des objets** : instanciation, modification d'attributs et appel de méthodes.

#### **3. Algorithmique**

* **Récursivité** : définition, exemples classiques (factorielle, suite de Fibonacci).
* **Complexité algorithmique** : notation Big-O, analyse de la complexité d'algorithmes simples (recherche linéaire, tri à bulles, tri rapide).
* **Structures de données** : listes, piles, files, dictionnaires, ensembles.
* **Algorithmes de tri et de recherche** : tri à bulles, tri rapide, recherche linéaire et dichotomique.

#### **4. Bibliothèques Python pour l’analyse des données (NumPy, Pandas)**

* **NumPy** : tableaux multidimensionnels, indexation, manipulation de données (slicing, broadcasting).
* **Pandas** : création de DataFrame, gestion des données manquantes, fusion de DataFrames, groupement et agrégation des données.
* **Manipulation et analyse de données** : tri, filtrage, nettoyage de données, calcul des statistiques descriptives.

#### **5. Bases de données : Modélisation conceptuelle et relationnelle**

* **Modélisation conceptuelle** : modèle entité-association (MERISE), diagrammes de classes.
* **Modélisation relationnelle** : normalisation, relations, clés primaires et étrangères, dépendances fonctionnelles.
* **Conception de schémas relationnels** : conversion du modèle conceptuel en schéma relationnel.

#### **6. SQL : Création, manipulation et interrogation des données**

* **Création de bases de données** : commandes CREATE, définition des tables et types de données.
* **Manipulation des données** : INSERT, UPDATE, DELETE.
* **Requêtes SQL** : sélection des données avec SELECT, filtrage avec WHERE, jointures (INNER JOIN, LEFT JOIN), sous-requêtes.
* **Agrégation et tri des données** : GROUP BY, HAVING, ORDER BY, fonctions d'agrégation (COUNT, AVG, SUM).
* **Optimisation des requêtes** : indexation, gestion des performances de requêtes complexes.

#### **7. Introduction à Power BI : Connexion, transformation, visualisation**

* **Connexion aux sources de données** : importer des données depuis des fichiers Excel, bases de données SQL, services Web.
* **Transformation des données** : nettoyage des données, ajout de colonnes calculées, fusion et agrégation des jeux de données.
* **Visualisation interactive** : création de graphiques (barres, lignes, camemberts), cartes géographiques, tableaux croisés dynamiques.
* **Création de tableaux de bord** : interactions entre visuels, filtres, rapports interactifs.

#### **8. Introduction à l’apprentissage profond avec PyTorch**

* **Concepts de base de l’apprentissage profond** : réseaux de neurones, fonctions d’activation, propagation avant et rétropropagation.
* **PyTorch** : création de réseaux de neurones, gestion des tenseurs, définition des modèles, entraînement avec des données.
* **Apprentissage supervisé et non supervisé** : classification, régression, clustering avec PyTorch.

#### **9. Traitement d’images avec OpenCV**

* **Introduction à OpenCV** : chargement, affichage et manipulation des images.
* **Filtrage d'images** : flou, détection de bords (Canny), transformation de couleurs (RGB, HSV).
* **Détection d'objets** : algorithmes de détection de contours, reconnaissance de formes, détection de visages.
* **Traitement avancé des images** : segmentation, transformation géométrique (rotation, mise à l'échelle).

### ****Compétences visées :****

* **Maîtriser Python** pour la programmation de données, avec une compréhension approfondie des bibliothèques NumPy et Pandas.
* **Utiliser la POO** pour structurer efficacement des projets de programmation.
* **Manipuler des bases de données relationnelles** à travers SQL, de la création à l’interrogation de données.
* **Créer des visualisations interactives** avec Power BI, et générer des rapports dynamiques.
* **Appliquer des concepts d’apprentissage profond** pour résoudre des problèmes pratiques via PyTorch.
* **Traiter des images** avec OpenCV pour extraire des informations pertinentes.
* Implémenter des modèles de deep learning simples.
* Appliquer des techniques de traitement d’images.

**Probabilités et Statistiques Appliquées**

**Volume horaire :**

* **Cours Magistral (CM)** : 24h
* **Travaux Dirigés (TD)** : 18h
* **Travaux Pratiques (TP)** : 12h

**Pré-requis :**  
Notions de base en mathématiques et en calcul différentiel. Une connaissance préalable des statistiques de base et des concepts de probabilité est un atout.

### ****Description générale :****

Ce cours introduit les concepts fondamentaux des probabilités et de la statistique, en se concentrant sur leur application pratique à l’analyse des données. Il présente les outils théoriques nécessaires à la modélisation des phénomènes aléatoires ainsi que les techniques statistiques pour effectuer des analyses inférentielles. Ce programme est essentiel pour les étudiants en sciences, en ingénierie et en science des données qui souhaitent comprendre et appliquer des méthodes statistiques sur des données réelles.

### ****Objectifs :****

* **Maîtriser les concepts fondamentaux** des probabilités et savoir modéliser des phénomènes aléatoires par des lois de probabilité adaptées.
* **Comprendre et appliquer** les lois de probabilité discrètes et continues, telles que la loi binomiale, de Poisson et normale.
* **Construire et interpréter** des indicateurs statistiques descriptifs pour résumer des ensembles de données.
* **Appliquer des méthodes d’estimation** (ponctuelle et par intervalle) et des **tests d’hypothèses** à des jeux de données réels.
* **Initier les étudiants à l’analyse de corrélation, de régression**, et à l’ajustement de modèles statistiques simples pour comprendre les relations entre variables.

### ****Plan du cours détaillé :****

#### **1. Statistique descriptive : mesures de tendance centrale, de dispersion, visualisation des données**

* **Mesures de tendance centrale** : moyenne, médiane, mode, propriétés et interprétations.
* **Mesures de dispersion** : écart-type, variance, étendue, interquartiles, coefficient de variation.
* **Visualisation des données** : histogrammes, diagrammes en boîte (boxplots), nuages de points, courbes de densité.
* **Résumé statistique des données** : table de fréquences, distributions empiriques.

#### **2. Espaces probabilisés : événements, axiomes de Kolmogorov, opérations sur les événements**

* **Concepts de base des probabilités** : événements, espace probabilisé, axiomes de Kolmogorov.
* **Opérations sur les événements** : réunion, intersection, différence, complément.
* **Probabilité conditionnelle** et **indépendance** des événements, formule de Bayes.

#### **3. Probabilités conditionnelles et indépendance**

* **Probabilité conditionnelle** : définition, interprétation, loi de probabilité conditionnelle.
* **Indépendance des événements** : critères d’indépendance, application au calcul de probabilités conditionnelles.
* **Formule de Bayes** : révision et applications dans des contextes réels, calcul de probabilités inverses.

#### **4. Variables aléatoires discrètes et continues : loi binomiale, loi de Poisson, loi normale, etc.**

* **Variables aléatoires discrètes** : lois de probabilité discrètes, exemple de la loi binomiale et de la loi de Poisson.
* **Variables aléatoires continues** : fonction de densité, loi normale, loi uniforme.
* **Propriétés des lois de probabilité** : espérance, variance, moments.

#### **5. Espérance, variance, moments, fonctions de répartition**

* **Espérance** : définition, propriétés et calcul pour des variables discrètes et continues.
* **Variance et écart-type** : définition, interprétation, calcul pour des variables discrètes et continues.
* **Moments d’une distribution** : premier et deuxième moment, application à la détermination de la forme des distributions.
* **Fonction de répartition** : définition, propriétés, calcul et utilisation pour déterminer des probabilités.

#### **6. Théorèmes limites : loi des grands nombres, théorème central limite**

* **Loi des grands nombres** : définition, interprétation, applications pratiques dans les séries temporelles.
* **Théorème central limite (TCL)** : formulation du théorème, convergence en distribution, implications pour les échantillons et estimations.
* **Applications du TCL** : approximations normales dans les grands échantillons, utilisation pour les tests statistiques.

#### **7. Estimation ponctuelle et par intervalle**

* **Estimation ponctuelle** : estimation de la moyenne, de la variance, de la proportion, propriétés des estimateurs.
* **Estimation par intervalle** : intervalle de confiance pour la moyenne et la proportion, méthodes de calcul.
* **Erreur d’estimation et précision** : impact de la taille de l’échantillon, compréhension du compromis entre précision et coût.

#### **8. Tests d’hypothèses : test de Student, Khi², tests sur proportions**

* **Concept des tests d’hypothèses** : formulation d’une hypothèse nulle et alternative, niveau de significativité.
* **Test de Student** : test t pour une moyenne, pour la différence de moyennes, applications en échantillons appariés et indépendants.
* **Test Khi²** : test d’ajustement et d’indépendance, calcul des degrés de liberté, applications sur des tableaux de contingence.
* **Tests sur proportions** : tests pour une proportion, comparaison de deux proportions.

#### **9. Corrélation, régression linéaire simple**

* **Corrélation** : coefficient de corrélation de Pearson, interprétation de la force et de la direction de la relation.
* **Régression linéaire simple** : modèle de régression, estimation des paramètres (méthode des moindres carrés), interprétation des coefficients.
* **Évaluation du modèle** : coefficient de détermination (R²), tests d’ajustement du modèle, diagnostic des résidus.

#### **10. Initiation aux méthodes statistiques avec Python (bibliothèques scipy.stats, statsmodels)**

* **Introduction aux bibliothèques Python** : utilisation de scipy.stats, statsmodels pour les tests statistiques.
* **Estimation et tests avec Python** : calcul des intervalles de confiance, tests t et Khi², régression linéaire.
* **Visualisation des résultats** : tracé des distributions, graphiques de régression et de corrélation avec matplotlib et seaborn.
* **Manipulation des données** : prétraitement des données avec Pandas pour les analyses statistiques.

### ****Compétences visées :****

* **Analyse descriptive des données** à l’aide d’outils graphiques et numériques, pour identifier les tendances et anomalies.
* **Modélisation probabiliste des phénomènes aléatoires** en choisissant les lois de probabilité appropriées et en analysant les événements aléatoires.
* **Interpréter les résultats statistiques** pour prendre des décisions éclairées basées sur les données.
* **Appliquer des tests d’hypothèses** dans un contexte pratique pour valider des conjectures sur des populations.
* **Implémenter des méthodes statistiques simples avec Python** pour effectuer des analyses sur des jeux de données réels.
* **Comprendre la régression et l'inférence statistique** pour analyser les relations entre les variables et faire des prévisions.

**Introduction à l’Algorithmique pour l’Intelligence Artificielle**

**Description générale :**  
Ce cours introductif a pour but de familiariser les étudiants avec les concepts de base en algorithmique appliqués à l’intelligence artificielle. Il couvre les structures de données essentielles, les notions de complexité, et les principaux types d’algorithmes utilisés pour la résolution de problèmes liés à l’IA. L’accent est mis sur la compréhension des mécanismes fondamentaux et leur mise en œuvre en Python.

**Objectifs :**

* Acquérir une culture de base en algorithmique utile à l’intelligence artificielle.
* Comprendre les structures de données simples et leur usage.
* Apprendre à concevoir et analyser des algorithmes élémentaires.
* Initier les étudiants aux algorithmes de recherche et de résolution de problèmes.
* Poser les bases pour les cours avancés d’apprentissage automatique et d’optimisation.

**Heures d’enseignement :**

* **Cours Magistral (CM)** : 15h
* **Travaux Dirigés (TD)** : 12h
* **Travaux Pratiques (TP)** : 9h

**Plan du cours :**

1. **Introduction à l’algorithmique et à la complexité**
   * Définitions, notations, mesure de performance
2. **Structures de données élémentaires**
   * Listes, tableaux, piles, files
3. **Algorithmes de tri et de recherche**
   * Tri à bulles, insertion, recherche linéaire et binaire
4. **Introduction aux graphes**
   * Représentation, parcours en largeur et en profondeur
5. **Notions de récursivité et d’itération**
   * Exemples classiques, bonne utilisation de la récursion
6. **Premiers pas vers l’IA**
   * Problèmes de recherche simple (labyrinthes, puzzles)
   * Algorithmes gloutons simples
7. **Implémentation Python**
   * Traduction des algorithmes en code clair et commenté

**Compétences visées :**

* Décrire et mettre en œuvre des algorithmes simples en Python.
* Choisir la structure de données appropriée pour un problème donné.
* Analyser la complexité d’un algorithme élémentaire.
* Résoudre de petits problèmes d’IA à l’aide de techniques algorithmiques de base.
* Préparer le terrain pour des cours plus avancés en intelligence artificielle et en science des données.

**Machine learning**

**Volume horaire total : 48 heures**

* **Cours magistral (CM) : 12h**  
  Présentation théorique des méthodes d’analyse de données multivariée (ACP, AFC, ACM, CAH, AFD, AFM, méthodes de partitionnement) et des principaux algorithmes de Machine Learning supervisé et non supervisé. Introduction aux outils logiciels utilisés (Python, R, SAS).
* **Travaux dirigés (TD) : 12h**  
  Études de cas et exercices guidés sur des jeux de données réels. Application des méthodes d’analyse multivariée (ACP, AFD, etc.) et classification (k-means, CAH) avec interprétation des résultats.
* **Travaux pratiques (TP – Projet encadré) : 24h**  
  Réalisation en groupe d’un projet de bout en bout : formulation du problème, analyse exploratoire, modélisation ML, validation, présentation finale.

**Pré-requis :**  
Statistiques de base, programmation en Python (NumPy, Pandas, Matplotlib).

**Description générale :**  
Ce module combine apprentissage théorique et mise en pratique à travers un projet en groupe basé sur des données réelles. Il articule les approches classiques d’analyse de données multivariée et la mise en œuvre de modèles de Machine Learning avec Python, R ou SAS. Les étudiants mèneront une démarche complète allant de la compréhension métier à la communication des résultats.

**Objectifs pédagogiques :**

* Identifier les enjeux d’un problème réel et structurer une démarche d’analyse.
* Maîtriser les méthodes classiques d’analyse multivariée : ACP, AFC, ACM, AFD, AFM.
* Utiliser les méthodes de classification hiérarchique et non hiérarchique.
* Implémenter des modèles de Machine Learning adaptés au problème.
* Maîtriser les outils Python, R ou SAS pour la data science.
* Restituer oralement les résultats de manière synthétique et pertinente.

**Contenu du module :**

1. **Introduction au projet, aux jeux de données et aux objectifs.**
2. **Réduction de dimension :**
   * Analyse en Composantes Principales (ACP)
   * Analyse Factorielle des Correspondances (AFC, ACM)
3. **Méthodes de classification :**
   * Classification Ascendante Hiérarchique (CAH)
   * Méthodes de partitionnement (k-means, etc.)
   * Analyse Factorielle Discriminante (AFD)
4. **Analyse de tableaux multiples (AFM).**
5. **Implémentation en Python/R/SAS :**
   * Prétraitement, visualisation, modélisation, validation croisée.
6. **Présentation orale finale :**
   * Démarche analytique, résultats, interprétation, recommandations.

**Compétences visées :**

* Conduite autonome d’une étude statistique et d’un projet de Machine Learning.
* Mise en œuvre de méthodes statistiques avancées pour l’analyse de données.
* Maîtrise des outils de programmation pour l’analyse (Python, R).
* Restitution synthétique d’un projet scientifique en équipe.

**Données : Modélisation, Qualité et Visualisation**

**Description générale :**  
Ce cours introduit les fondements de la gestion des données dans une perspective orientée IA et science des données. Il couvre la modélisation des données (conceptuelle et logique), les enjeux de qualité des données (cohérence, complétude, nettoyage) et les techniques modernes de visualisation pour l’analyse exploratoire. Le cours alterne des apports théoriques et des applications pratiques en Python et Power BI.

**Objectifs :**

* Comprendre comment structurer et modéliser les données pour un usage analytique.
* Appréhender les dimensions de la qualité des données et les techniques de traitement des données manquantes ou erronées.
* Savoir choisir et produire des visualisations adaptées aux types de données et aux objectifs d’analyse.
* Développer une culture critique autour de l’usage des données dans l’IA.

**Heures d’enseignement :**

* **Cours Magistral (CM)** : 15h
* **Travaux Dirigés (TD)** : 12h
* **Travaux Pratiques (TP)** : 9h

**Plan du cours :**

1. **Introduction générale**
   * Rôle des données dans l’IA et la science des données
   * Types de données : structurées, semi-structurées, non structurées
2. **Modélisation des données**
   * Modèle conceptuel (diagrammes Entité-Relation)
   * Modèle logique relationnel et transformation
   * Normalisation et intégrité référentielle
3. **Qualité des données**
   * Dimensions de la qualité : complétude, cohérence, exactitude, unicité
   * Détection et traitement des données manquantes ou aberrantes
   * Déduplication et validation des données
4. **Nettoyage et préparation des données**
   * Techniques avec Pandas (Python)
   * Recodage, filtrage, transformation
   * Fusion de jeux de données hétérogènes
5. **Visualisation des données – Fondamentaux**
   * Principes de perception visuelle et choix graphiques
   * Types de visualisations : histogrammes, courbes, heatmaps, boxplots, etc.
   * Bonnes pratiques pour éviter les biais visuels
6. **Outils et cas pratiques**
   * Visualisation avec Matplotlib, Seaborn, Plotly
   * Initiation à Power BI pour les tableaux de bord interactifs
   * Étude de cas : analyse exploratoire d’un jeu de données public

**Compétences visées :**

* Concevoir un modèle conceptuel de données adapté à un besoin fonctionnel.
* Nettoyer et préparer un jeu de données pour l’analyse.
* Évaluer la qualité d’un jeu de données et corriger les défauts courants.
* Construire des visualisations informatives, pertinentes et esthétiques.
* Manipuler des bibliothèques Python et des outils de BI pour la représentation des données.

**Pré-requis :**

* Bases en Python et connaissance élémentaire des structures de données.
* Aucune connaissance avancée en statistiques n’est requise.

**Préparation au TOEFL (Anglais Professionnel)**

**Description :**  
Ce cours prépare les étudiants à l’examen TOEFL iBT en renforçant leurs compétences en compréhension écrite et orale, expression écrite et orale, dans un contexte académique et professionnel.

**Objectifs :**

* Comprendre le format du TOEFL iBT
* Maîtriser les stratégies pour chaque section de l’examen
* Améliorer le vocabulaire et la grammaire académiques
* S’exprimer efficacement à l’oral et à l’écrit

**Heures d’enseignement :**

* CM : 12h | TD : 18h | TP : 10h

**Plan du cours :**

1. Introduction au TOEFL
2. Reading : stratégies de lecture critique
3. Listening : prise de notes et compréhension
4. Speaking : structuration et fluidité
5. Writing : essai indépendant et intégré
6. Vocabulaire et grammaire clés
7. Examens blancs et feedback

**Compétences visées :**

* Lire, écouter, écrire et parler efficacement en anglais académique
* Réussir le test TOEFL avec une bonne gestion du temps et des stratégies ciblées

**SEMESTRE 8**

**Recherche opérationnelle**

**Volume horaire :**

* **Cours Magistral (CM)** : 24h
* **Travaux Dirigés (TD)** : 16,5h
* **Travaux Pratiques (TP)** : 16h

**Pré-requis :**

* Optimisation convexe, bases en mathématiques et algorithmique.

### ****Description générale :****

Le cours de **Recherche Opérationnelle** explore les méthodologies et techniques mathématiques permettant de résoudre des problèmes concrets à travers des approches d'optimisation discrète. Les étudiants apprendront à appliquer ces techniques à des problèmes réels et à les implémenter en utilisant des outils de programmation comme Python ou Matlab. À la fin du cours, ils seront capables de modéliser et résoudre des problèmes complexes de recherche opérationnelle en utilisant des méthodes exactes ou approchées.

### ****Objectifs :****

* **Maîtriser les techniques de base en recherche opérationnelle**, telles que l'optimisation sous contraintes, la programmation par contraintes, et l'optimisation combinatoire.
* **Apprendre à modéliser des problèmes mathématiquement** en les traduisant en termes d'optimisation et en choisissant les méthodes appropriées pour les résoudre.
* **Utiliser la programmation linéaire** pour résoudre des problèmes en appliquant la méthode du simplexe et l'analyse de sensibilité.
* **Étudier les méthodes d'optimisation combinatoire** (méthodes énumératives, programmation dynamique, programmation linéaire en nombres entiers).
* **Appliquer les techniques de résolution** en analysant la complexité des algorithmes et en choisissant entre méthodes exactes ou approchées selon le contexte du problème.
* **Explorer des applications pratiques** dans des domaines tels que l’ordonnancement, le transport, et la planification.

### ****Plan du cours détaillé :****

#### **1) Théorie des graphes**

* **Introduction aux graphes** : définitions fondamentales, types de graphes (graphe orienté, non orienté, pondéré).
* **Représentation des graphes** : matrices d’adjacence, listes d’adjacence, arbres.
* **Propriétés des graphes** : connectivité, cycles, bipartition.

#### **2) Codage des graphes**

* **Représentation et stockage des graphes** : matrices et listes d'adjacence, complexité en espace et temps.
* **Codage des graphes dans des programmes** : conversion entre différentes représentations.
* **Manipulation des graphes en programmation** : ajout, suppression de sommets et arêtes, recherche de voisins.

#### **3) Composantes connexes et parcours**

* **Composantes connexes** : définition et identification des composantes connexes dans un graphe.
* **Parcours en largeur (BFS)** et **parcours en profondeur (DFS)** : algorithmes et applications.
* **Applications** : détection de cycles, recherche de connexité dans des réseaux.

#### **4) Algorithmes de plus courts chemins entre deux sommets (Dijkstra) et entre tous les sommets (Floyd)**

* **Algorithme de Dijkstra** : principe, complexité, applications.
* **Algorithme de Floyd-Warshall** : principe et application à la recherche des plus courts chemins entre tous les sommets.
* **Optimisation de l'algorithme de Dijkstra** : utilisation de structures de données adaptées (tas de Fibonacci).

#### **5) Problème du voyageur de commerce**

* **Problème du voyageur de commerce (TSP)** : formulation et applications.
* **Algorithme glouton** : solution approximative, analyse de la performance.
* **Méthode du recuit simulé** : principe, paramètres, et applications.
* **Algorithme par colonie de fourmis** : approche inspirée de la nature, principes, et applications.

#### **6) Programmation linéaire et méthode du simplexe**

* **Introduction à la programmation linéaire** : formulation de problèmes d’optimisation sous contraintes linéaires.
* **Méthode du simplexe** : algorithme, principe, critères d'arrêt, et applications pratiques.
* **Dualité en programmation linéaire** : interprétation géométrique, théorème de dualité, sensibilité.
* **Analyse de sensibilité** : étude des variations des solutions optimales par rapport aux changements dans les paramètres du problème.

#### **7) Méthodes de flot dans un graphe**

* **Problèmes de flot dans un réseau** : définition, applications dans la logistique et les transports.
* **Algorithme de Ford-Fulkerson** : principe, calcul du flot maximum.
* **Problème du flot minimal-cost** : formulation, méthodes de résolution, applications.

### ****Compétences visées :****

* **Maîtriser les concepts fondamentaux** de la recherche opérationnelle pour résoudre des problèmes d'optimisation discrète.
* **Programmer des algorithmes d'optimisation** en utilisant des outils comme Python et Matlab pour appliquer des méthodes exactes et approchées.
* **Modéliser des problèmes complexes** en optimisation discrète, tels que le problème du voyageur de commerce ou le problème de flot, et les résoudre efficacement.
* **Choisir et appliquer la méthode d'optimisation appropriée** en fonction du type de problème et des contraintes (ex. programmation linéaire, programmation dynamique, recuit simulé).
* **Appliquer les méthodes de recherche opérationnelle** dans des domaines variés tels que l'ordonnancement, la logistique, la planification et le transport.

### Modèles de régression

**Volume horaire :**

* Cours magistral (CM) : 18h
* Travaux dirigés (TD) : 12h
* Travaux pratiques (TP) : 22h

**Pré-requis :**  
Statistiques descriptives, notions de probabilités, bases en programmation (Python ou R), initiation à l’analyse multivariée.

### ****Description :****

Ce module couvre l’ensemble du cycle de traitement des données : de l’exploration initiale à la modélisation avancée. Il combine des techniques classiques d’analyse exploratoire et factorielle, de régression linéaire et logistique, de modèles mixtes, et de plans d’expériences. L'accent est mis sur l’interprétation des résultats et la mise en œuvre via Python, R ou SAS.

### ****Objectifs pédagogiques :****

* Maîtriser les fondements des méthodes exploratoires, descriptives et prédictives.
* Savoir modéliser une variable quantitative ou catégorielle via des approches linéaires.
* Être capable d'analyser des données expérimentales à l’aide de plans adaptés.
* Implémenter les méthodes à l’aide d’outils statistiques modernes (Python, R, SAS).
* Interpréter les résultats et proposer des recommandations fiables.

### ****Plan du cours :****

#### **Partie 1 – Analyse exploratoire et multivariée (8h CM / 4h TD / 6h TP)**

1. Nettoyage, visualisation, exploration univariée et bivariée
2. ACP, AFC, ACM
3. Classification : CAH, k-means, méthode de partitionnement
4. Analyse factorielle discriminante, tableaux multiples (AFM)

#### **Partie 2 – Modèles de régression linéaire et logistique (7h CM / 4h TD / 8h TP)**

1. Régression linéaire simple et multiple (moindres carrés, Gauss-Markov, ANOVA)
2. Sélection de variables : AIC, BIC, méthodes forward/backward
3. Régression sur données corrélées : PLS, PCR, Ridge, Lasso
4. Régression logistique, modèle probit, estimation MLE
5. Évaluation des modèles : déviance, résidus, sensibilité/spécificité, courbes ROC

#### **Partie 3 – Modèles mixtes et plans d’expériences (3h CM / 4h TD / 8h TP)**

1. Modèles à effets fixes et aléatoires, estimation REML
2. Plans factoriels, blocs, split-plot, mesures répétées
3. Diagramme de Hasse, confusion d’effets, plans fractionnaires
4. Surfaces de réponse : plans de Doehlert, Box-Behnken, D-optimaux

### ****Compétences visées :****

* Concevoir une analyse statistique robuste sur des données réelles.
* Appliquer des modèles de régression adaptés selon les types de variables.
* Utiliser des plans expérimentaux pour optimiser les résultats.
* Maîtriser les bibliothèques de data science : pandas, statsmodels, scikit-learn, ggplot2, SAS.
* Restituer oralement ou par écrit une étude de données rigoureuse.

**Séries Temporelles**

#### **Description du Cours**

Ce cours est une introduction approfondie aux séries chronologiques. Il explore les concepts fondamentaux tels que la tendance, la saisonnalité, les moyennes mobiles, le lissage exponentiel et la prévision Holt-Winters. Il couvre également les processus stochastiques, y compris les processus autorégressifs (AR), les processus de moyenne mobile (MA), ainsi que les modèles combinés ARMA et ARIMA.

#### **Objectifs**

* Comprendre et modéliser l'évolution d'un phénomène temporel à l’aide d’approches déterministes (tendance et saisonnalité) et stochastiques (processus ARMA et analyse des résidus).
* Appliquer des méthodes de lissage et de prévision pour générer des prédictions précises à partir de données chronologiques.
* Analyser les caractéristiques des séries chronologiques afin de développer des modèles adaptés aux données observées.

#### **Heures d'enseignement**

* **Cours Magistral (CM)** : 24 heures
* **Travaux Dirigés (TD)** : 15 heures
* **Travaux Pratiques (TP)** : 16 heures

#### **Pré-requis Obligatoires**

* Notions de statistiques descriptives et de probabilités : moyenne, variance, covariance de variables aléatoires, indépendance.

#### **Plan du Cours**

1. **Introduction aux Séries Chronologiques**
2. **Estimation de la Tendance et de la Saisonalité**
3. **Décomposition d’une Série Chronologique**
4. **Prévision par Lissage Exponentiel**
5. **Outils pour l’Étude des Processus Stationnaires**
6. **Processus Linéaires MA et AR**
7. **Processus ARMA et ARIMA**

#### **Compétences Visées**

* **Modélisation déterministe** : Savoir modéliser la partie déterministe d’une série chronologique, notamment la tendance et la saisonnalité.
* **Modélisation stochastique** : Utiliser les processus AR, MA et ARMA pour modéliser la partie aléatoire d’une série chronologique.
* **Prédiction** : Appliquer les méthodes de prévision, y compris le lissage exponentiel et les modèles ARMA, pour effectuer des prédictions fiables.

**Martingales et chaînes de Markov**

**Volume horaire :**

* **Cours Magistral (CM)** : 12h
* **Travaux Dirigés (TD)** : 9h
* **Travaux Pratiques (TP)** : 8h

**Pré-requis :**

* Probabilités de niveau Licence 3 (espaces probabilisés, variables aléatoires, lois).
* Notions d'intégration et d'analyse fonctionnelle.

### ****Description générale :****

Ce cours constitue une introduction approfondie aux **martingales** et aux **chaînes de Markov** à temps discret. Il met l’accent sur des notions fondamentales comme l’espérance conditionnelle, les propriétés de convergence des martingales, et le comportement à long terme des chaînes de Markov. Ces concepts sont essentiels pour aborder des cours probabilistes et statistiques avancés, et permettent de comprendre et de modéliser des processus stochastiques dans de nombreux domaines, notamment en finance, en biologie, et en intelligence artificielle.

### ****Objectifs :****

* **Comprendre et manipuler l'espérance conditionnelle**, une notion centrale dans la théorie des martingales et des chaînes de Markov.
* **Construire et étudier des martingales à temps discret**, en analysant leur comportement et leurs propriétés de convergence.
* **Maîtriser les chaînes de Markov à temps discret**, comprendre leur dynamique, et étudier leur comportement asymptotique à long terme.
* **Appliquer les théorèmes fondamentaux** des martingales et des chaînes de Markov dans des contextes pratiques (simulations, applications probabilistes).

### ****Plan du cours détaillé :****

#### **1) Espérance conditionnelle**

* **Définitions de l'espérance conditionnelle** : pour un processus stochastique donné, la notion d'espérance conditionnelle conditionnée par une information partielle.
* **Propriétés de l'espérance conditionnelle** : linéarité, loi de l'itération des espérances, théorème de la loi totale de probabilité.
* **Calculs d'espérances conditionnelles** dans des contextes simples : variables discrètes et continues, exemples pratiques.
* **Applications** : utilisation de l'espérance conditionnelle dans les martingales et les chaînes de Markov.

#### **2) Martingales**

* **Définition d'une martingale** : notion de filtrage, conditions de martingales (égalité entre espérance conditionnelle et la valeur passée).
* **Temps d'arrêt** : définition, importance et application dans les martingales.
* **Théorème d’arrêt des martingales** : énoncé, preuve et applications pratiques.
* **Décomposition de Doob** : décomposition d'une martingale en une somme de martingales et processus de type intégré.
* **Convergence des martingales** : critères de convergence, théorèmes fondamentaux de convergence (exemple : théorème de convergence de Doob).
* **Applications pratiques** : étude et simulation de martingales simples dans des contextes financiers ou autres.

#### **3) Chaînes de Markov**

* **Définition et propriétés des chaînes de Markov** : processus de Markov, matrices de transition, et indépendance conditionnelle.
* **Équation de Chapman-Kolmogorov** : formulation et compréhension de l’évolution des probabilités de transition dans une chaîne de Markov.
* **Classification des états** : récurrence, transience, et états absorbants.
* **Matrices de transition et classes d'équivalence** : analyse de la structure des chaînes de Markov à travers les classes d’états.
* **Propriétés à long terme des chaînes de Markov** : existence de mesures invariantes et comportement stationnaire.
* **Applications pratiques** : modélisation et simulation de chaînes de Markov dans des systèmes simples.

#### **4) Mesures invariantes et théorèmes ergodiques**

* **Mesures invariantes** : définition et existence dans les chaînes de Markov.
* **Théorème ergodique** : énoncé et application à la convergence vers une distribution stationnaire.
* **Applications en statistiques** : estimation de paramètres à partir de chaînes de Markov.

#### **5) Illustrations numériques en TP**

* **Simulation de chaînes de Markov** : génération de trajectoires de chaînes de Markov à partir de matrices de transition.
* **Simulation de martingales simples** : étude numérique de martingales simples à travers des exemples pratiques.
* **Analyse de convergence** : observation du comportement asymptotique de chaînes de Markov et de martingales via des simulations.
* **Applications pratiques** : résoudre des problèmes en utilisant Python ou Matlab (ex : modélisation d’un système de file d'attente, jeu de hasard).

### ****Compétences visées :****

* **Calculer et interpréter les espérances conditionnelles** dans des contextes de martingales et chaînes de Markov.
* **Construire et étudier des martingales à temps discret**, en analysant leurs propriétés de filtrage, leur convergence et leur comportement.
* **Simuler des processus de Markov et des martingales simples** à l’aide d’outils numériques comme Python ou Matlab.
* **Appliquer les théorèmes fondamentaux des martingales et des chaînes de Markov** pour analyser le comportement asymptotique des processus stochastiques et résoudre des problèmes pratiques en probabilités et statistiques.

**Équations aux dérivées partielles et discrétisation**

**Volume horaire :**

* **Cours Magistral (CM)** : 24h
* **Travaux Dirigés (TD)** : 15h
* **Travaux Pratiques (TP)** : 16h

**Pré-requis :**

* Théorie des EDO
* Calcul différentiel
* Notions de base en analyse fonctionnelle

### ****Description générale :****

Ce cours se concentre sur l'étude des **équations aux dérivées partielles (EDP)** et leur discrétisation, en mettant l'accent sur la compréhension et la modélisation des phénomènes physiques à travers ces équations. Nous commencerons par explorer des exemples classiques d'EDP et les rappels nécessaires en calcul différentiel. Le cours aborde les différentes classifications des EDP (par non-linéarité, ordre, et type). Il présente également des méthodes analytiques et numériques pour résoudre ces équations, telles que la méthode des caractéristiques pour les EDP du premier ordre, les formulations variationnelles et les approches de discrétisation pour des applications numériques. Ce cours constitue un fondement essentiel pour les étudiants souhaitant appliquer les EDP à des problèmes pratiques en sciences et ingénierie.

### ****Objectifs :****

* **Se familiariser avec les EDP** utilisées pour décrire des phénomènes physiques tels que la diffusion, la propagation des ondes, et les problèmes de transport.
* **Introduire les concepts de solution classique**, problème bien-posé, et l’importance de ces notions pour garantir l'existence et l'unicité des solutions des EDP.
* **Apprendre à construire et représenter les solutions** des EDP classiques à travers des méthodes analytiques et numériques (par exemple, séparation des variables, transformée de Fourier).
* **Explorer la notion de formulation variationnelle** et son application à la résolution des EDP, avec des exemples concrets de modélisation et de discrétisation numérique.

### ****Plan du cours détaillé :****

#### **Introduction : Notion d’EDP et modélisation**

* **Exemples classiques d’EDP** : Équations de transport, d’ondes, et de Laplace.
* **Problème bien-posé** : Conditions de Cauchy, conditions aux bords, et importance des solutions classiques et non classiques.
* **Classification des EDP** : Par ordre, type (elliptique, parabolique, hyperbolique), et linéarité (linéaire, semi-linéaire, non-linéaire).

#### **Chapitre I : EDP d’ordre un et méthode des caractéristiques**

* **Équations linéaires et semi-linéaires** : Introduction aux équations d’ordre un et leur résolution par les caractéristiques.
* **Équations quasilinéaires et non-linéaires** : Approches et généralisation à Rn\mathbb{R}^nRn.
* **Méthode des caractéristiques** : Détail des étapes pour résoudre les EDP du premier ordre et applications à des problèmes physiques.

#### **Chapitre II : Loi de conservation et solution faible**

* **Introduction à la loi de conservation** : Concept de conservation de la masse, de l'énergie, et d'autres quantités physiques.
* **Solution faible** : Définition et application à des équations de type hyperbolique.
* **Relation de Rankine-Hugoniot** : Application à la résolution de l’équation de Burgers et compréhension des solutions entropiques.

#### **Chapitre III : Autres approches de représentation des solutions**

* **Équation des ondes et de la chaleur** : Techniques classiques de séparation des variables pour les EDP du second ordre.
* **Transformation de Fourier** : Utilisation de la transformée de Fourier pour résoudre les EDP de la chaleur et autres équations aux dérivées partielles.
* **Applications pratiques** : Résolution de problèmes en physique en utilisant ces méthodes analytiques.

#### **Chapitre IV : Équations elliptiques et formulation variationnelle**

* **Équations elliptiques** : Problèmes de type Laplace et Poisson, propriétés de régularité des solutions.
* **Analyse hilbertienne** : Présentation des espaces de Hilbert et leur rôle dans les formulations des EDP.
* **Théorème de Lax-Milgram** : Existence et unicité des solutions d’EDP elliptiques par la méthode variationnelle.
* **Formulation variationnelle** : Résolution des EDP à travers une approche faible, et analyse des résultats via des méthodes numériques.

### ****Compétences visées :****

* **Analyser les modèles des phénomènes physiques** modélisés par des EDP, et comprendre les conditions d’existence des solutions.
* **Résoudre des EDP classiques** en utilisant des méthodes analytiques telles que la méthode des caractéristiques et la séparation des variables.
* **Appliquer les formulations variationnelles** pour résoudre des EDP, en particulier dans les contextes elliptiques.
* **Mettre en œuvre des techniques de discrétisation** pour résoudre des EDP numériquement et analyser les résultats des algorithmes numériques ou des logiciels spécialisés.

**Modélisation numérique**

**Volume horaire :**

* **Cours Magistral (CM)** : 12h
* **Travaux Dirigés (TD)** : 9h
* **Travaux Pratiques (TP)** : 8h

**Pré-requis :**

* Connaissances mathématiques de niveau Licence 3 (analyse, algèbre, EDO)

### ****Description générale :****

Ce cours se concentre sur la formalisation de problèmes issus de diverses disciplines telles que la **physique**, l'**écologie**, et l'**économie**, sous forme de modèles mathématiques. L'accent est mis sur l'application de méthodes numériques pour résoudre ces modèles à l'aide de logiciels de calcul scientifique comme **Python**, **Matlab**, ou **Scilab**. Les étudiants apprendront à construire et analyser des modèles à la fois discrets et continus, en mettant l'accent sur la simulation et l'optimisation des systèmes dynamiques. Ce cours est conçu pour les étudiants qui souhaitent utiliser les outils mathématiques et informatiques pour résoudre des problèmes complexes du monde réel.

### ****Objectifs :****

* **Acquérir les bases de la formalisation mathématique** d'un problème réel et comprendre comment le traduire en un modèle mathématique adapté.
* **Construire et analyser des modèles** discrets et continus, en se concentrant sur leur dynamique et les résultats obtenus.
* **Mettre en œuvre des méthodes numériques** adaptées à la simulation de systèmes dynamiques, en utilisant des outils numériques comme Python, Matlab ou Scilab.
* **Explorer les techniques d'optimisation** dans la modélisation pour améliorer les performances des systèmes simulés.

### ****Plan du cours détaillé :****

#### **1. Introduction à la modélisation numérique – Exemples issus des sciences appliquées**

* Présentation générale des principes de la **modélisation numérique** dans les domaines de la physique, de l'écologie et de l'économie.
* Étude de cas pratiques pour illustrer l'importance de la modélisation dans la résolution de problèmes réels.
* Discussion des différentes approches et outils utilisés pour la simulation numérique des systèmes complexes.

#### **2. Étude de systèmes récurrents linéaires**

* Introduction aux systèmes dynamiques **linéaires**.
* Analyse des **systèmes récurrents linéaires**, résolution numérique par les **méthodes d'intégration**.
* Étude des **propriétés de stabilité**, de la convergence et de l'impact des conditions initiales sur la solution.

#### **3. Modèles non linéaires : points fixes, stabilité, bifurcation**

* Exploration des **modèles non linéaires** avec un focus sur les **points fixes** et leur stabilité.
* Introduction aux concepts de **bifurcation** et de **chaos** dans les systèmes dynamiques.
* Utilisation de méthodes numériques pour analyser ces phénomènes dans des modèles réels.

#### **4. Simulation numérique de systèmes dynamiques continus**

* **Simulation numérique** de systèmes continus : équations différentielles et leurs solutions numériques.
* Introduction aux méthodes d'intégration comme **Euler**, **Runge-Kutta**, et d'autres techniques pour résoudre des systèmes dynamiques.
* Application à des modèles physiques, biologiques et économiques pour observer la dynamique des systèmes continus.

#### **5. Introduction à l’optimisation dans la modélisation**

* Introduction aux **méthodes d'optimisation** appliquées à la modélisation numérique.
* Utilisation de **techniques d'optimisation** pour améliorer la précision des simulations ou pour résoudre des problèmes de décision dans les modèles économiques et écologiques.
* Mise en œuvre d'algorithmes d'optimisation classiques : **descente de gradient**, **algorithmes génétiques**, et autres approches pour la recherche de solutions optimales.

#### **6. Travaux Pratiques (TP) : Simulation de comportements dynamiques – Cas en écologie, économie, physique**

* Application pratique des concepts du cours à travers des **simulations** de systèmes dynamiques dans des cas d'études réels issus de l'écologie, de l'économie et de la physique.
* Utilisation de **Python**, **Matlab** ou **Scilab** pour coder et simuler les modèles dynamiques.
* Interprétation des résultats des simulations et discussion sur les implications des comportements observés dans les systèmes réels.

### ****Compétences visées :****

* **Maîtriser les méthodes de résolution numérique** pour les systèmes dynamiques, en appliquant des techniques adaptées à la continuité ou la discrétisation des problèmes.
* **Modéliser et simuler des phénomènes réels** à travers des modèles mathématiques et des simulations numériques en utilisant des outils comme Python, Matlab ou Scilab.
* **Construire des algorithmes adaptés à la résolution de modèles** spécifiques, en tenant compte des particularités des systèmes modélisés.
* **Interpréter les résultats numériques** dans un cadre applicatif, en analysant les comportements et en tirant des conclusions pratiques.
* **Travailler au sein d’équipes pluridisciplinaires** sur des projets de simulation, en mettant en œuvre des techniques collaboratives pour résoudre des problèmes complexes dans des domaines variés.

**Deep Learning**

**Volume horaire :**

* **Cours Magistral (CM)** : 12h
* **Travaux Dirigés (TD)** : 9h
* **Travaux Pratiques (TP)** : 8h

**Pré-requis :**

* Connaissances de base en **machine learning**
* Programmation en **Python**
* Notions de **calcul matriciel** et **optimisation**

### ****Description générale :****

Ce cours offre une introduction approfondie aux concepts, techniques et applications du **Deep Learning**, une branche du machine learning qui se concentre sur l'utilisation de réseaux neuronaux complexes pour résoudre des problèmes d'intelligence artificielle. Les étudiants exploreront les **architectures de réseaux neuronaux modernes** et apprendront à les implémenter en **Python** avec des bibliothèques comme **TensorFlow** et **PyTorch**. L'accent est mis sur l'acquisition d'une solide **compréhension théorique**, tout en pratiquant les concepts à travers des **projets concrets** appliqués à différents domaines tels que la **vision par ordinateur**, les **séries temporelles** et le **traitement du langage naturel (NLP)**.

### ****Objectifs :****

* **Comprendre les principes fondamentaux** du **Deep Learning** et des principales architectures de réseaux neuronaux.
* **Implémenter des modèles de réseaux neuronaux** pour des cas d’usage dans des domaines tels que **vision par ordinateur**, **séries temporelles** et **NLP (Natural Language Processing)**.
* **Utiliser les bibliothèques Python** standards du domaine, telles que **TensorFlow** et **PyTorch**, pour concevoir et entraîner des modèles.
* **Optimiser** les modèles en utilisant des techniques de régularisation, d’entraînement et d'évaluation adaptées.

### ****Plan du cours détaillé :****

#### **1. Introduction au Deep Learning et aux réseaux de neurones artificiels**

* Présentation du **Deep Learning** : concepts de base et motivation derrière l’utilisation de réseaux neuronaux pour l’intelligence artificielle.
* Historique des réseaux de neurones et évolution vers des architectures profondes.
* Architecture de base d’un **réseau de neurones artificiels (ANN)** et comparaison avec des modèles classiques de machine learning.

#### **2. Fonctionnement des réseaux : propagation avant, backpropagation, descente de gradient**

* Explication du processus de **propagation avant** et de **backpropagation**.
* Algorithme de **descente de gradient** et ses variantes pour l'entraînement des réseaux neuronaux.
* Introduction à l’optimisation : **fonction de coût**, **taux d'apprentissage**, et **convergence**.

#### **3. Réseaux convolutifs (CNN) pour l’analyse d’images**

* Introduction aux **réseaux convolutifs (CNN)** : principes de **convolution**, **pooling**, et **activation**.
* Applications des CNN en **vision par ordinateur**, reconnaissance d'images et classification.
* Mise en œuvre des CNN en Python avec TensorFlow et PyTorch pour résoudre des tâches pratiques d'analyse d'images.

#### **4. Réseaux récurrents (RNN, LSTM) pour les données séquentielles**

* Étude des **réseaux récurrents (RNN)** et de leur capacité à traiter des **données séquentielles** (comme des séries temporelles et des séquences textuelles).
* Introduction aux **LSTM (Long Short-Term Memory)** et leur amélioration par rapport aux RNN classiques.
* Applications des RNN et LSTM pour des tâches comme la prédiction de séries temporelles et le traitement du langage naturel.

#### **5. Introduction aux transformers et au NLP**

* Présentation des **transformers**, une architecture avancée pour le traitement du langage naturel.
* Concepts de **self-attention** et **mécanisme d'attention**.
* Application des transformers au **NLP (Natural Language Processing)** pour des tâches comme la traduction automatique, l’analyse de texte, et la génération de texte.

#### **6. Entraînement, régularisation et évaluation de modèles**

* Techniques de **régularisation** pour éviter le **surapprentissage** (overfitting), telles que **dropout**, **L2 regularization**, etc.
* **Évaluation des modèles** : métriques de performance (précision, rappel, F1-score, etc.) et validation croisée.
* **Optimisation avancée** : Adam, RMSprop, et autres techniques d'optimisation adaptées au Deep Learning.

#### **7. Projets pratiques en vision, séries temporelles et texte**

* Application des techniques abordées dans des **projets concrets** : classification d'images, prédiction de séries temporelles, et analyse de texte en utilisant des réseaux de neurones.
* Mise en œuvre complète d’un projet de **Deep Learning** : de la collecte des données à l’entraînement du modèle et à l’évaluation des performances.
* Utilisation des **frameworks TensorFlow** et **PyTorch** pour implémenter et entraîner des modèles sur des datasets réels.

### ****Compétences visées :****

* **Concevoir, entraîner et évaluer des réseaux de neurones** adaptés à différents types de données (images, séries temporelles, texte).
* **Identifier les architectures pertinentes** en fonction de la nature du problème et de l’application (CNN, RNN, LSTM, transformers).
* **Utiliser efficacement TensorFlow et PyTorch** pour développer des solutions d’IA dans des domaines variés.
* **Appliquer les meilleures pratiques** d’entraînement et d’optimisation pour obtenir des modèles performants et généralisables.
* **Intégrer le Deep Learning dans des projets concrets**, en tirant parti des bibliothèques Python pour la mise en œuvre pratique de solutions d'IA.

### ****Statistiques Bayésienne et Paramétriques****

**Volume horaire :**

* Cours magistral (CM) : 16h
* Travaux dirigés (TD) : 8h
* Travaux pratiques (TP) : 16h

**Pré-requis :**  
Probabilités, statistique inférentielle classique, bases en programmation (Python ou R), connaissance de distributions usuelles.

### ****Description :****

Ce module introduit les fondements des deux approches principales de l’inférence statistique : l’approche **paramétrique fréquentiste** (estimations ponctuelles et intervalles de confiance) et l’approche **bayésienne** (actualisation des croyances via les lois de probabilité). Les concepts sont illustrés par des applications concrètes en Python ou R.

### ****Objectifs pédagogiques :****

* Comprendre et comparer les paradigmes fréquentiste et bayésien.
* Maîtriser les méthodes d’estimation et de tests dans un cadre paramétrique.
* Modéliser l’incertitude et construire des distributions a posteriori en statistique bayésienne.
* Implémenter des méthodes bayésiennes simples (MCMC, loi a priori conjugée).
* Interpréter correctement les résultats selon chaque cadre d’analyse.

### ****Plan du cours :****

#### **Partie I – Statistique Paramétrique (8h CM / 4h TD / 6h TP)**

1. Rappels : modèles paramétriques, vraisemblance, statistique suffisante
2. Estimateurs : moments, vraisemblance, moindres carrés
3. Propriétés des estimateurs : biais, variance, efficacité (théorèmes de Cramér-Rao, Rao-Blackwell)
4. Tests paramétriques : tests de Student, de Wald, de la vraisemblance, du score
5. Intervalles de confiance et estimation par bootstrap

#### **Partie II – Statistique Bayésienne (8h CM / 4h TD / 10h TP)**

1. Paradigme bayésien : loi a priori, vraisemblance, loi a posteriori
2. Lois a priori non informatives, informatives, et conjuguées
3. Cas usuels : loi normale, binomiale, poisson (modèles conjugés)
4. Estimation bayésienne (MAP, espérance a posteriori)
5. Méthodes de simulation : introduction à MCMC, échantillonnage de Gibbs, Metropolis-Hastings
6. Comparaison Bayésien / Fréquentiste : choix de modèle, intervalle de crédibilité vs. de confiance

### ****Outils pratiques :****

* Python (NumPy, SciPy, PyMC, ArviZ)
* R (rstan, brms, bayesplot)

### ****Compétences visées :****

* Construire un modèle statistique paramétrique cohérent.
* Appliquer une démarche bayésienne de manière rigoureuse.
* Réaliser des inférences à partir de simulations (notamment MCMC).
* Comparer les méthodes selon les contextes (données rares, incertitude forte…).
* Communiquer les résultats dans un cadre d’aide à la décision.

### ****Systèmes de recommandation****

**Volume horaire :**

* Cours magistral (CM) : 12h
* Travaux dirigés (TD) : 8h
* Travaux pratiques (TP) : 16h

**Pré-requis :**  
Notions de machine learning, statistiques, programmation en Python (Pandas, NumPy, Scikit-learn), bases de données.

### ****Description :****

Ce module présente les fondements théoriques et les techniques pratiques des **systèmes de recommandation** utilisés dans les moteurs de recherche, e-commerce, plateformes de streaming ou réseaux sociaux. Il couvre les approches collaboratives, content-based et hybrides, avec des mises en œuvre concrètes en Python.

### ****Objectifs pédagogiques :****

* Comprendre les principes et enjeux des systèmes de recommandation.
* Implémenter les principales approches de recommandation : filtrage collaboratif, filtrage basé sur le contenu, modèles hybrides.
* Évaluer les performances des systèmes de recommandation.
* Intégrer des systèmes de recommandation dans une chaîne de traitement de données.

### ****Plan du cours :****

#### **1. Introduction générale (2h CM / 1h TD)**

* Définitions et cas d’usage
* Données implicites vs explicites
* Problèmes courants : sparsité, froid démarrage, biais

#### **2. Méthodes basées sur le contenu (2h CM / 1h TD / 3h TP)**

* Représentation des utilisateurs et des items
* Similarité cosinus, TF-IDF, distances
* Recommandation personnalisée via profil utilisateur

#### **3. Filtrage collaboratif (3h CM / 2h TD / 5h TP)**

* Approche user-based et item-based
* Matrices d’évaluation, k-NN, SVD, factorisation matricielle
* Implémentation avec Surprise ou Scikit-learn

#### **4. Modèles hybrides et avancés (3h CM / 2h TD / 5h TP)**

* Méthodes hybrides : pondérées, combinées, méta-niveaux
* Intégration de facteurs temporels, contextuels, démographiques
* Introduction aux réseaux de neurones pour la recommandation (deep learning, auto-encodeurs)

#### **5. Évaluation des systèmes de recommandation (2h CM / 2h TP)**

* Métriques : RMSE, MAE, précision, rappel, F1, MAP, NDCG
* Méthodologies de validation : k-fold, leave-one-out

### ****Compétences visées :****

* Choisir et justifier une approche de recommandation adaptée à un contexte
* Implémenter des algorithmes simples et avancés en Python
* Évaluer rigoureusement la qualité d’un système de recommandation
* Interpréter les résultats et proposer des améliorations

### ****Outils mobilisés :****

* Python (Pandas, NumPy, Scikit-learn, Surprise, TensorFlow/Keras)
* Librairies de visualisation (Matplotlib, Seaborn)
* Jeux de données : MovieLens, Amazon, GoodBooks, etc.

**Droit et Éthique des Données**

**Description :**  
Ce cours explore les aspects juridiques et éthiques du traitement des données personnelles, avec un focus sur le RGPD et les défis éthiques des technologies modernes comme l’IA et le big data.

**Objectifs :**  
• Comprendre les principes fondamentaux du droit des données personnelles.  
• Analyser les enjeux éthiques du traitement des données.  
• Appliquer le RGPD à des cas pratiques.

**Heures d’enseignement :**  
• Cours magistral (CM) : 12h  
• Travaux dirigés (TD) : 9h

**Pré-requis :**  
Connaissances de base en droit et informatique.

**Plan du cours :**

1. **Introduction au Droit des Données**
   * Concepts clés et cadre juridique global (RGPD, CCPA).
2. **Le RGPD et ses Principes**
   * Droits des individus, obligations des entreprises.
3. **Éthique et Confidentialité**
   * Enjeux éthiques : vie privée, biais algorithmique.
4. **Sécurité des Données**
   * Mesures de sécurité, gestion des violations.
5. **Responsabilité et Conformité**
   * Responsabilité des entreprises, sanctions juridiques.
6. **Études de Cas**
   * Analyse de cas réels de non-conformité et bonnes pratiques.

**Compétences visées :**

• Appliquer le RGPD dans des situations réelles.  
• Identifier et discuter des dilemmes éthiques dans la gestion des données.

**SEMESTRE 9**

**Optimisation**

**Description :**  
Ce cours introduit les fondements de l’optimisation, à la fois du point de vue théorique et algorithmique. Il aborde les grandes classes de problèmes, les conditions d’optimalité, ainsi que les méthodes numériques de résolution avec des applications concrètes en MATLAB.

**Objectifs :**  
• Comprendre les bases théoriques de l’optimisation avec ou sans contraintes.  
• Connaître et utiliser les principaux algorithmes d’optimisation.  
• Appliquer ces méthodes à des problèmes concrets.

**Heures d’enseignement :**  
• Cours magistral (CM) : 24h  
• Travaux dirigés (TD) : 15h  
• Travaux pratiques (TP) : 16h

**Pré-requis :**  
• Calcul différentiel en dimension n.

**Plan du cours :**

1. **Formulation d’un problème d’optimisation**
2. **Convexité et existence de solution**
3. **Conditions d’optimalité (avec et sans contraintes)**
   * Multiplicateurs de Lagrange, théorème de Kuhn-Tucker
4. **Algorithmes numériques**
   * Gradient, Newton, quasi-Newton, pénalisation, Uzawa
5. **Implémentation sous MATLAB**
6. **Introduction aux méthodes globales**
   * Recuit simulé, algorithmes génétiques

**Compétences visées :**

• Modéliser un problème d’optimisation.  
• Choisir et implémenter un algorithme adapté.  
• Interpréter les résultats et les appliquer à des cas concrets.

**Système dynamique et contrôle**

**Description :**  
Étude de l'évolution temporelle de systèmes dynamiques et introduction aux bases du contrôle automatique.

**Objectifs :**

* Modéliser les systèmes dynamiques continus et discrets.
* Concevoir des contrôleurs simples pour stabiliser les systèmes.

**Heuresd'enseignement :**

* CM : 12h
* TD : 9h
* TP : 8h

**Pré-requis obligatoires :**  
Analyse appliquée, calcul différentiel.

**Plan du cours :**

1. Systèmes dynamiques linéaires et non-linéaires.
2. Stabilité et points fixes.
3. Introduction à la théorie du contrôle.
4. Contrôle PID.
5. Applications en ingénierie et économie.

**Compétencesvisées :**

* Analyser et simuler des systèmes dynamiques.
* Concevoir des stratégies de contrôle simples.

**Modélisation aléatoire et simulation stochastique**

**Description :**  
Ce cours est une introduction approfondie à la méthode de Monte Carlo. Il traite des techniques de simulation de variables aléatoires, des méthodes de réduction de variance et des algorithmes de Monte Carlo pour les chaînes de Markov, incluant notamment l'algorithme de Hastings-Metropolis et l'algorithme de Gibbs.

**Objectifs :**

* Approcher une espérance et construire un intervalle de confiance.
* Simuler des variables aléatoires de lois diverses.
* Mettre en œuvre des algorithmes stochastiques avancés pour la simulation de chaînes de Markov.

**Heuresd’enseignement :**

* Coursmagistral (CM) : 24h
* Travauxdirigés (TD) : 16,5h
* Travauxpratiques (TP) : 16h

**Pré-requisobligatoires :**

* Probabilités de niveau Licence (espace probabilisé, variables aléatoires, espérance, variance, théorèmes limites)

**Plan du cours :**

1. Rappels de probabilités et introduction à la méthode de Monte Carlo
2. Simulation de variables aléatoires : méthode d’inversion, méthode du rejet
3. Techniques de réduction de variance : échantillonnage préférentiel, variables antithétiques, variables de contrôle
4. Méthodes de Monte Carlo pour les chaînes de Markov : algorithmes de Hastings-Metropolis, de Gibbs, recuit simulé

**Compétencesvisées :**

* Simuler des variables aléatoires de lois complexes.
* Approcher des espérances par des méthodes de Monte Carlo.
* Estimer des intervalles de confiance pour des quantités simulées.
* Mettre en œuvre des algorithmes de simulation stochastique pour des chaînes de Markov.

**MLOps&DEVOps**

A revoir …………………………..

**Description :**  
Découverte des bonnes pratiques pour automatiser, déployer et maintenir des solutions de Machine Learning.

**Objectifs :**

* Comprendre l’automatisation du cycle de vie du machine learning.
* Déployer des modèles de manière fiable et reproductible.

**Heures d'enseignement :**

* CM : 12h
* TD : 9h
* TP : 8h

**Pré-requis obligatoires :**  
Notions de machine learning et programmation.

**Plan du cours :**

1. Introduction à DevOps et MLOps.
2. CI/CD appliqué aux projets ML.
3. Docker, Kubernetes pour la mise en production.
4. Pipelines ML avec MLflow.
5. Surveillance et maintenance de modèles déployés.

**Compétences visées :**

* Déployer des modèles ML à l’échelle industrielle.
* Automatiser les workflows de data science.

### ****Apprentissage pour les séries temporelles****

**Volume horaire :**

* Cours magistral (CM) : 12h
* Travaux dirigés (TD) : 8h
* Travaux pratiques (TP) : 16h

**Pré-requis :**  
Notions de séries chronologiques (tendance, saisonnalité, stationnarité), statistiques de base, Python (Pandas, Scikit-learn), bases du Machine Learning (régression, classification, overfitting, etc.).

### ****Description :****

Ce module explore l’application des méthodes d’apprentissage automatique à l’analyse, la modélisation et la prévision de séries temporelles. Il couvre les approches classiques et modernes, allant des modèles statistiques régularisés jusqu’aux réseaux de neurones récurrents et convolutionnels.

### ****Objectifs pédagogiques :****

* Comprendre les spécificités des séries temporelles pour l’apprentissage automatique.
* Préparer des données temporelles pour la modélisation.
* Maîtriser différentes approches de prédiction séquentielle supervisée et non supervisée.
* Évaluer et comparer les performances des modèles sur des données temporelles.

### ****Plan du cours :****

#### **1. Introduction à l’apprentissage temporel (2h CM / 1h TD)**

* Spécificités des données temporelles
* Préparation des features temporelles (lag, fenêtre glissante, encodage du temps)
* Formulation de tâches de prédiction (next-step, multi-step)

#### **2. Modèles de régression appliqués aux séries (2h CM / 1h TD / 3h TP)**

* Régression linéaire, régularisation (Ridge, Lasso)
* Forêts aléatoires et boosting (XGBoost, LightGBM)
* Feature engineering temporel

#### **3. Apprentissage profond pour séries temporelles (3h CM / 2h TD / 5h TP)**

* Réseaux de neurones simples pour séries
* Réseaux récurrents (RNN, LSTM, GRU)
* Réseaux convolutionnels 1D (CNN)
* Encodage séquentiel et attention

#### **4. Modèles séquentiels avancés (3h CM / 1h TD / 4h TP)**

* Séquence à séquence (Seq2Seq)
* Transformers pour séries temporelles
* Modèles hybrides (statistique + deep learning)

#### **5. Évaluation des modèles temporels (2h CM / 2h TP)**

* Backtesting, validation glissante, horizon fixe vs variable
* Métriques : RMSE, MAPE, sMAPE, MAE
* Visualisation des performances

### ****Compétences visées :****

* Préparer et structurer des données temporelles pour l’apprentissage automatique
* Choisir une approche adaptée (modèle classique ou deep learning)
* Implémenter, entraîner et évaluer des modèles temporels en Python
* Interpréter les résultats et ajuster les hyperparamètres
* Appliquer ces méthodes à des données réelles (prévision de ventes, séries financières, IoT…)

### ****Outils mobilisés :****

* Python (Pandas, Scikit-learn, XGBoost, Keras/TensorFlow, PyTorch)
* Librairies spécialisées : statsmodels, tslearn, Darts, GluonTS, sktime
* Données : météo, ventes, trafic web, consommation électrique, etc.

### ****Apprentissage par renforcement****

**Volume horaire :**

* Cours magistral (CM) : 12h
* Travaux dirigés (TD) : 8h
* Travaux pratiques (TP) : 16h

**Pré-requis :**  
Notions d’algèbre linéaire, probabilités, Python (NumPy, Matplotlib), introduction à l’apprentissage supervisé.

### ****Description :****

Ce module introduit les concepts fondamentaux de l’apprentissage par renforcement (Reinforcement Learning - RL), un paradigme dans lequel un agent apprend à interagir avec un environnement afin de maximiser une récompense. Le cours couvre les méthodes tabulaires, l’approximation de fonction, les méthodes de contrôle et l’apprentissage profond par renforcement.

### ****Objectifs pédagogiques :****

* Comprendre les bases théoriques de l’apprentissage par renforcement.
* Maîtriser les algorithmes classiques de RL (Q-learning, SARSA, Monte Carlo).
* Implémenter des agents simples dans des environnements simulés.
* Appliquer des techniques de Deep RL (DQN, Policy Gradient).
* Évaluer et optimiser les politiques d’un agent.

### ****Plan du cours :****

#### **1. Introduction au RL et processus de décision de Markov (MDP) (2h CM / 1h TD)**

* Environnement, agent, états, actions, récompenses
* Notion de politique, valeur d’un état, équation de Bellman
* Exemples d’application : jeux, robotique, systèmes de recommandation

#### **2. Méthodes tabulaires (3h CM / 2h TD / 3h TP)**

* Algorithmes Monte Carlo (avec et sans importance sampling)
* SARSA et Q-learning
* Exploration vs exploitation (ε-greedy, softmax)

#### **3. Approximation de fonction (2h CM / 1h TD / 3h TP)**

* Problème de grande dimension
* Approximation linéaire, fonctions de base, features
* Algorithmes TD(λ)

#### **4. Deep Reinforcement Learning (3h CM / 2h TD / 6h TP)**

* Deep Q-Networks (DQN) : replay buffer, target networks
* Policy Gradient, REINFORCE
* Avantages et limites des méthodes basées sur réseaux de neurones

#### **5. Applications et environnements simulés (2h CM / 1h TD / 4h TP)**

* Gymnasium / OpenAI Gym
* Études de cas : navigation, bandits, jeux simples (CartPole, FrozenLake)
* Visualisation et évaluation des politiques

### ****Compétences visées :****

* Comprendre et formuler un problème de RL en termes de MDP
* Implémenter et tester des algorithmes de RL tabulaires et profonds
* Évaluer la performance d’un agent et optimiser sa stratégie
* Utiliser des bibliothèques Python de RL (Gym, Stable-Baselines3, PyTorch)

### ****Outils mobilisés :****

* Python (NumPy, Matplotlib, PyTorch ou TensorFlow)
* Librairies : gymnasium, stable-baselines3, RLlib
* Environnements simulés : jeux, environnements dynamiques, robots virtuels