CentraleSupélec

Catalogue de cours

Diplôme d'Ingénieur Informatique

Mention Sciences des Données et de l'Information (Metz)

Campus de Metz de CentraleSupélec

dernière mise à jour : 28 avril 2026

Semestre 09

SD9		**SD9**	**8 ECTS**
3MD1520	50	Programmation avancée en C++	21.0 h
3MD1540	50	Apprentissage automatique 1	27.5 h
3MD1510	50	Ingénierie des applications logicielles	22.5 h
3MD1530	50	Modèles statistique 1	29.0 h

SM10		**SM10**	**8 ECTS**
3MD4040	50	Apprentissage profond	28.5 h
3MD4010	50	Apprentissage automatique 2	23.0 h
3MD4050	50	Modèles statistiques 2	33.0 h
3MD4030	*40*	*HPC-HPDA sur GPU en CUDA*	*22.5 h*
3MD4020	*40*	*Modèles parcimonieux*	*21.5 h*

Semestre 10

SM11		**SM11**	**8 ECTS**
3MD4120	40	Apprentissage par renforcement	23.0 h
3MD4150	50	Traitement automatique du langage naturel	29.0 h
3MD4140	40	Apprentissage statistique	23.0 h
3MD4130	40	Modèles de calcul du Big Data	21.0 h
3MD4110	40	Perspectives en Apprentissage et Intelligence Artificielle	23.5 h

Programmation avancée en C++

Responsable de cours : Hervé Frezza-Buet, Frédéric Pennerath

Total : 21.0 h **TP :** 18.0 h

3MD1520 *retour*

Description : Savoir coder efficacement un algorithme dans un langage de programmation donné suppose comprendre préalablement le modèle de calcul associé et la façon dont les instructions de ce langage sont traduites en instructions machine. Trop d'étudiants abordent encore la programmation de façon superficielle et hasardeuse, faute de connaître les notions de base nécessaires à l'écriture d'un code élégant et performant. La force singulière du langage C++ est de permettre la production de codes compilés proches du code machine optimal tout en offrant différentes approches de programmation de haut-niveau comme le typage fort, la programmation objet, la programmation fonctionnelle et la méta-programmation (génération automatique de code à la compilation). Pour cette raison, C++ est devenu le langage incontournable pour développer des algorithmes optimisés. Son seul inconvénient tient à sa richesse qui n'a cessé de croître dans ses versions les plus récentes (C++11/14/17/20) et qui rend le langage difficile à appréhender dans sa totalité sans formation adéquate. Ce cours est destiné aux étudiants, y compris débutants, désireux de maîtriser les différents aspects de la programmation en C++ pour pouvoir écrire du code qui concilie performance et élégance. Le cours adopte une approche bottom-up en partant des mécanismes d'exécution de programmes élémentaires pour aller progressivement vers les fonctionnalités les plus avancées du langage.

Contenu : 6 séances de TP (3h chacunes), et un examen individuel sur machine (2h). Le programme porte sur la chaîne de compilation, la gestion mémoire (pile et tas), les types, l'héritage, la généricité (templates), la programmation système.

Prérequis : Savoir programmer (boucles, fonctions, rudiments d'approche objet) dans un langage informatique, avoir des rudiments d'architecture des ordinateurs (processeur, mémoire, etc.).

Acquis d'apprentissage : Savoir écrire un programme en C++ utilisant différents paradigmes de programmation comme la programmation objet, la programmation fonctionnelle et la programmation générique. Connaître certains aspects du langage C++ qui ont une influence déterminante sur les performances des programmes lors de leur exécution. Connaître les fonctionnalités offertes par les spécifications les plus récentes du langage C++ (C++11, C++14, C++17, C++20). Savoir utiliser un environnement de compilation et de débogage C++

Méthodes pédagogiques : L'objectif est de transmettre aux étudiants un réel savoir-faire de la programmation, d'une part en illustrant les notions à travers des exemples d'utilisation pertinents, d'autre part en consacrant une part importante du volume horaire aux travaux de laboratoire.

Moyens : TP sous PC Linux / g++ 40 étudiants max. 2 étudiants par machine max.

Modalités d'évaluation : Épreuve individuelle de 3h sur machine, rattrapable.

Compétences évaluées :
- Être opérationnel, responsable et innovant dans le monde numérique

---

TP :

1. Introduction (3.0 h)
2. Héritage (3.0 h)
3. Approche fonctionnelle (3.0 h)
4. Patrons (3.0 h)
5. Programmation système (3.0 h)
6. Structures de données de la STL (3.0 h)

Apprentissage automatique 1

Responsable de cours : Hervé Frezza-Buet

Total : 27.5 h **CM :** 16.5 h, **TP :** 9.0 h

3MD1540 *retour*

Description : Ce cours donne le cadre général de l'apprentissage automatique, permettant de situer les différentes approches du domaine. Sont abordés les notions de prétraitement des données, une introduction à la théorie de l'apprentissage statistique (risques, sur-apprentissage, proxys convexes, régularisation), la différence entre approches fréquentistes et bayésiennes, les paradigmes d'apprentissage supervisé, non-supervisé, semi-supervisé, et par renforcement. Quelques approches sont détaillées (méthodes à Noyaux, SVM, Boosting, Bagging, Arbres de décision...).

Acquis d'apprentissage : À l'issue de ce cours, les élèves sauront reconnaître dans le paysage des nombreuses méthodes disponibles sur l'étagère les différentes classes d'algorithmes. Ils auront de plus les notions statistiques qui leur permettront un usage raisonné de ces méthodes, évitant ainsi une approche boîte noir avec essai des paramètres à l'aveugle.

Moyens : Les cours et travaux de pratiques sont donnés par Hervé Frezza-Buet, Arthur Hoarau, Jérémy Fix. Les cours présentent les aspects théoriques, des preuves mathématiques, mais sont également illustrés par des démonstrations des algorithmes. Les travaux pratiques s'effectueront en python, en utilisant sickit-learn, en binômes.

Modalités d'évaluation : Examen écrit de 2h, rattrapable.

Compétences évaluées :
- Analyser, concevoir et réaliser des systèmes complexes
- Développer ses compétences dans un domaine d'ingénieur et dans un métiers

---

CM :

1. Datasets and learning (1.5 h)
2. Frequentist, Bayesian, evaluation (1.5 h)
3. Risks (1.5 h)
4. C-SVC, Lagrange formulation (1.5 h)
5. Kernels, numerical resolution (1.5 h)
6. SVMs for regression, unsupervized learning, nu-versions of SVMs. (1.5 h)
7. Intro supervisé (1.5 h)
8. Arbres de décision (1.5 h)
9. Réduction de dim. (1.5 h)
10. Partitionnement/Regrouppement (1.5 h)
11. Quantification vectorielle (1.5 h)

TP :

1. Data Science en Python (3.0 h)
2. Arbres de décision (3.0 h)
3. Non Supervisé (3.0 h)

Ingénierie des applications logicielles

Responsable de cours : Virginie Galtier

Total : 22.5 h **CM :** 6.0 h, **TD :** 7.5 h, **TP :** 9.0 h

3MD1510 *retour*

Description : L'architecture applicative d'un système définit ses composants logiciels et les flux qu'ils échangent, ainsi que leur implantation sur l'architecture matérielle. Elle se trouve naturellement à la croisée des métiers de développement et d'exploitation. Ce cours dresse une typologie des principales architectures applicatives et présente les concepts du "devops", ensemble de pratiques facilitant l'automatisation de la livraison de logiciels. Quelques concepts seront mis en œuvre sur une architecture illustrative qui servira de fil rouge au cours.

Contenu : Introduction au devops (cycle de vie du logiciel, agilité, CI CD pipeline, build, versioning, tests, conteneurs, infrastructure as code) : principes et aperçu de quelques outils. Typologie des architectures applicatives et technologies associées, middleware (accent sur un MOM)

Prérequis : Avant ce cours les élèves doivent réviser leur cours de SIP de 1A. Ils doivent en outre être à l'aise en Java (des ressources préparatoires sont fournies), avec Git et Linux.

Acquis d'apprentissage : A l'issue de ce cours, les élèves seront capables de choisir l'architecture applicative la plus adaptée à leur projet et sauront mettre en œuvre une solution logicielle à base de file d'attente de messages. Ils pourront s'intégrer rapidement dans une chaîne de développement et de déploiement d'application et seront familiarisés avec l'utilisation de quelques outils.

Méthodes pédagogiques : Les concepts généraux seront présentés en cours. Des tutoriels pratiques et guidés permettront ensuite aux étudiants de se familiariser avec les approches et outils associés. Enfin ils mettront en application leurs nouvelles connaissances et compétences sur un projet fil rouge s'enrichissant au fil des séances et qui constituera un exemple pour le projet d'évaluation réalisé en autonomie.

Moyens : Les parties pratiques du cours utilisent des logiciels libres qui pourront être installés directement sur les ordinateurs personnels des étudiants. Un conteneur « clef-en-main » est également fourni. Les étudiants bénéficient des clusters du campus pour certains déploiements.

Modalités d'évaluation : L'acquisition des connaissances et compétences visées par ce cours sera évaluée sur la base de quelques tests écrits individuels au cours des séances de TP et d'un projet

Compétences évaluées :
- Être opérationnel, responsable et innovant dans le monde numérique

---

CM :

1. introduction, software application architecture (1.5 h)
2. application integration (1.5 h)
3. devops (1.5 h)
4. software quality (1.5 h)

TD :

1. Kafka, a distributed streaming platform (1.5 h)
2. Kafka, a distributed streaming platform (1.5 h)
3. Docker, a containerization solution (1.5 h)
4. Kubernetes, an orchestration system (1.5 h)
5. GitLab CI/CD, unit tests and SonarKube (1.5 h)

TP :

1. Wikipedia project with only Kafka (3.0 h)
2. Deploy on the cluster (3.0 h)
3. CI/CD of the Wikipedia project on GitLab (3.0 h)

Modèles statistique 1

Responsable de cours : Frédéric Pennerath

Total : 29.0 h **CM :** 16.5 h, **TD :** 4.5 h, **TP :** 6.0 h

3MD1530 *retour*

Description : Les cours "Modélisation statistique" ModStat1 et ModStat2 traitent de la modélisation de systèmes pour lesquels les sorties sont suffisamment incertaines pour qu'elles nécessitent d'être modélisées par des variables aléatoires. Le cours débute avec des rappels en statistique et l'introduction de modèles élémentaires (e.g. naive Bayes, régression linéaire, etc) pour aller progressivement vers des modèles plus complexes. Si les cours présentent les modèles et méthodes élémentaires les plus utiles dans ce contexte de modélisation, ils ne se veulent pas être un catalogue exhaustif. L'objectif est davantage de présenter au sein d'une théorie cohérente les concepts et outils théoriques communs à l'ensemble de ces modèles et méthodes et de montrer comment à partir d'hypothèses de modélisation propres à chaque problème concret, ces concepts sont assemblés logiquement avant d'aboutir à une méthode opérationnelle. D'un point de vue pratique l'enjeu de ce cours est non seulement de donner aux étudiants les moyens de comprendre et d'utiliser à bon escient les implémentations existantes de modèles mais aussi de concevoir ses propres implémentations pour prendre en compte les particularités d'un problème donné. Le cours s'attache à rattacher la théorie à la pratique : sont d'abord identifiées en cours les hypothèses associées à une classe de problèmes donnée, s'ensuit un travail théorique de modélisation, qui conduit à définir un modèle et ses algorithmes d'estimation. Ces résultats sont alors appliqués à un cas d'étude en TD avant de faire l'objet en TP d'un travail d'implémentation (en Python) et d'évaluation sur des données. Le cours ModStat1 introduira les outils de bases de la modélisation statistique quand le cours ModStat2 se concentrera sur les modèles à variables cachées.

Prérequis : - Connaissances de base en théorie des probabilités, statistique et apprentissage machine (machine learning) - Niveau débutant en programmation Python / Numpy

Acquis d'apprentissage : - Être capable de choisir un modèle/ une méthode statistique adaptée au problème considéré et de la mettre en œuvre de façon appropriée. - Être capable de comprendre les concepts théoriques sous-jacents à une méthode d'inférence statistique présentée dans un article scientifique. - Être capable d'implémenter un modèle / une méthode statistique dans un langage tel que Python. - Être capable d'adapter un modèle / une méthode pour tenir compte des spécificités du problème traité.

Modalités d'évaluation : Examen écrit de 2h sans documents, rattrapable.

Compétences évaluées :
- Analyser, concevoir et réaliser des systèmes complexes
- Développer ses compétences dans un domaine d'ingénieur et dans un métiers

---

CM :

1. Modèles statistiques (1.5 h)
2. Estimation (1.5 h)
3. Estimation bayésienne (1.5 h)
4. Réseaux bayésiens 1 (1.5 h)
5. Naive Bayes (1.5 h)
6. Réseaux bayésiens 2 (1.5 h)
7. Causalité (1.5 h)
8. Modèles gaussiens (1.5 h)
9. Modèles linéaires (1.5 h)
10. Famille exponentielle (1.5 h)
11. GLM (1.5 h)

TD :

1. Estimation bayésienne (1.5 h)
2. Modélisation causale (1.5 h)
3. Régression (1.5 h)

TP :

1. Modélisation et modèles gaussiens (3.0 h)
2. Régression (3.0 h)

Apprentissage profond

Responsable de cours : Jérémy Fix

Total : 28.5 h **CM :** 13.5 h, **TP :** 15.0 h

3MD4040 *retour*

Description : L'apprentissage profond est une technologie en plein essor ces dernières années, notamment grâce à l'utilisation des GPUs (Graphical Processing Units), la disponibilité de grandes masses de données mais aussi la compréhension d'éléments théoriques permettant de mieux définir des architectures de réseaux de neurones plus facilement entrainables. Dans ce cours, les étudiants seront introduits aux bases des réseaux de neurones et également aux différents éléments architecturaux qui permettent de concevoir un réseau de neurones en fonction du problème de prédiction considéré. Le cours est décomposé en modules dans lesquels on aborde les questions des algorithmes d'optimisation, leur initialisation, les techniques de régularisation, les architectures complètement connectées, les réseaux convolutifs, les réseaux récurrents, les techniques d'introspection. Des travaux pratiques sur GPUs sont associés aux cours.

Contenu : Les sujets abordés seront les suivants :
- Introduction historiques aux réseaux de neurones, classifieur/régresseur linéaire (1.5 HPE)
- Graphe de calcul et descente de gradient, Réseaux complètement connectés, RBFs, Auto-encodeurs, Méthodes d'optimisations, initialisation, régularisation (3 HPE)
- Réseaux convolutifs : architectures (1.5 HPE)
- Réseaux convolutifs : classification, détection d'objet, ségmentation sémantique (1.5 HPE)
- Réseaux récurrents : architectures et entrainement (1.5 HPE)
- Réseaux récurrents : applications (1.5 HPE)
- Modèles génératifs : modèles auto-regressifs, VAE, GANs et diffusion (1.5 HPE)

Les TP porteront sur :
- Introduction à pytorch par la classification avec des réseaux linéaires, complètements connectés et convolutifs (3 HPE)
- Réseaux convolutifs pour la ségmentation sémantique (3 HPE)
- Réseaux récurrents pour la conversion séquence à séquence (3 HPE)
- Réseaux génératifs adversiaux (3 HPE)
- Représentations Neuronales Implicites (3 HPE)

Prérequis : On supposera que les étudiants auront des connaissance en algèbre linéaire, en optimisation, en vision par ordinateur et en programmation python. Les étudiants doivent disposer d'une certaine aisance dans un environnement Linux.

Acquis d'apprentissage : Être capable d'implémenter et de déployer un algorithme de deep learning (prise en main des frameworks de deep learning et déploiement du calcul du GPU)
Être capable de choisir l'architecture de réseau de neurones adaptée au problème de prédiction considéré
Savoir diagnostiquer l'apprentissage d'un réseau de neurones (qu'apprends le réseau de neurone ? comment l'apprend t'il ? apprends t'il ? est-il capable de généraliser ?)

Méthodes pédagogiques : Le cours est construit autour d'un cours magistral pendant lequel les notions théoriques et expérimentales sont introduites et illustrées au travers d'exemple. Des TPs sont régulièrement répartis le long du cours pour pouvoir mettre en œuvre les notions vues en cours.

Moyens : Nous utiliserons le framework python Pytorch. Les étudiants pourront travailler en binôme et disposeront des GPUs du Data Center d'Enseignement du campus de Metz pour faire les TPs.
Une page dédidée sera créée sur eduano. Des forums de discussion seront ouvert pour permettre aux étudiants de poser leurs questions sur le cours et d'interagir entre eux et avec l'équipe enseignante.

Modalités d'évaluation : L'évaluation reposera sur deux éléments : des évaluations sur des questionnaires papier en début de chaque TP (1/3 de la note) et la participation, en équipe, à un challenge créé pour l'occasion (2/3 de la note). Les élèves devront également réaliser une présentation filmée décrivant leur approche et leurs résultats.

Compétences évaluées :
- Agir, entreprendre, innover en environnement scientifique et technologique
- Être opérationnel, responsable et innovant dans le monde numérique
- Penser et agir en ingénieur éthique, responsable et intègre

Ressources externes :

— Site du cours
— Sujets des TPs

CM :

1. Introduction et réseaux linéaires (1.5 h)
2. Graphe de calcul, descente de gradients et réseaux feedforward (1.5 h)
3. Optimisation, Initialisation, Régularisation (1.5 h)
4. Eléments d'architectures des réseaux convolutifs (1.5 h)
5. Applications des réseaux convolutifs à la détection d'objets et la segmentation sémantique (1.5 h)
6. Les approches par transformers (1.5 h)
7. Eléments d'architecture des réseaux récurrents (1.5 h)
8. Architectures et applications des réseaux récurrents (1.5 h)
9. Modèles génératifs : modèles auto-regressifs, VAE, GANs et diffusion (1.5 h)

TP :

1. Introduction à PyTorch par la classification d'images (3.0 h)
2. Réseaux convolutifs pour la segmentation sémantique (3.0 h)
3. Application des RNNs pour la retranscription de la parole (3.0 h)
4. Modèles génératifs (3.0 h)
5. Représentations Neurales Implicites (3.0 h)

Apprentissage automatique 2

Responsable de cours : Arthur Hoarau

Total : 23.0 h **CM :** 9.0 h, **TP :** 12.0 h

3MD4010 *retour*

Description : Ce cours complète le cours d'apprentissage automatique 1 avec les notions de traitement de données (réduction de dimension, etc.), l'apprentissage non-supervisé, l'apprentissage actif et semi-supervisé, les questions d'explicabilité.

Acquis d'apprentissage : À l'issue de ce cours, les élèves auront complété leur approche en largeur de l'apprentissage automatique.

Modalités d'évaluation : Examen écrit de 2h, rattrapable.

Compétences évaluées :
- Analyser, concevoir et réaliser des systèmes complexes
- Développer ses compétences dans un domaine d'ingénieur et dans un métiers

---

CM :

1. Bagging (1.5 h)
2. Boosting (1.5 h)
3. Détection anomalies (1.5 h)
4. Quantification Incertitude (1.5 h)
5. Apprentissage semi-supervisé (1.5 h)
6. Explicabilité (1.5 h)

TP :

1. Bagging (3.0 h)
2. Forêts aléatoires (3.0 h)
3. Détection anomalies/OOD (3.0 h)
4. semi-supervisé/XAI (3.0 h)

Modèles statistiques 2

Responsable de cours : Frédéric Pennerath

Total : 33.0 h **CM :** 12.0 h, **TD :** 6.0 h, **TP :** 12.0 h

3MD4050 *retour*

Description : Ce cours est le prolongement du cours ModStat1. Il s'articule autour des trois concepts fondamentaux de processus stochastique, de variable latente et de techniques d'inférence approchées. La première partie du cours sur les processus s'articule autour de trois grandes familles de processus : les processus ponctuels, les processus de Markov et les processus Gaussiens. La notion de variable latente est ensuite abordée à travers les modèles de mélange et l'algorithme EM. Les deux notions sont ensuite combinées pour développer les modèles de Markov cachés, dans le cas d'états discrets (HMM) comme continus (Filtres de Kalman). Enfin sont présentées les techniques d'inférence approchées avec d'une part les techniques d'échantillonnage (MCMC) et l'inférence variationnelle.

Prérequis : - Avoir suivi le cours "Modèles statistiques 1" - Niveau débutant en programmation Python / Numpy

Acquis d'apprentissage : A l'issue de ce cours, les élèves sauront associer à des séries de données le type de processus stochastique qui leur correspond et appliquer les méthodes d'estimation associées. Ils sauront également spécifier un modèle intégrant des variables cachées et appliquer l'algorithme EM pour en estimer les paramètres. Ils sauront modéliser un problème de clustering sous la forme d'un modèle de mélange. Ils sauront spécifier une HMM ou un filtre de Kalman pour modéliser le comportement dynamique d'un système à état discret ou continu.

Modalités d'évaluation : Examen écrit de 3h avec documents, rattrapable.

Compétences évaluées :
- Analyser, concevoir et réaliser des systèmes complexes
- Développer ses compétences dans un domaine d'ingénieur et dans un métiers

---

CM :

1. Processus ponctuels (1.5 h)
2. Processus de Markov (1.5 h)
3. Processus gaussiens (1.5 h)
4. Modèles de mélanges (1.5 h)
5. Modèles de Markov cachés (1.5 h)
6. Filtre de Kalman (1.5 h)
7. Echantillonnage (1.5 h)
8. Inférence variationnelle (1.5 h)

TD :

1. Processus de Poisson et de Markov (1.5 h)
2. Modèles de Markov cachés (1.5 h)
3. Filtre de Kalman (1.5 h)
4. Echantillonnage (1.5 h)

TP :

1. Processus gaussiens (3.0 h)
2. Modèles de mélange (3.0 h)
3. Filtre de Kalman et particul. (3.0 h)

4. Echantillonnage (3.0 h)

HPC-HPDA sur GPU en CUDA

Responsable de cours : Stéphane Vialle

Total : 22.5 h (électif) **CM :** 6.0 h, **TD :** 4.5 h, **TP :** 12.0 h

3MD4030 *retour*

Description : Ce cours a pour objectif de présenter l'algorithmique et la programmation à haute performance sur GPU, avec des mises en oeuvre sur des algorithmes de Machine Learning, et sur des plates-formes de calculs équipées de GPU.

Contenu : Architecture des GPU Principes algorithmiques du parallélisme de données à grain fin sur GPU (modèles SIMD et SIMT) Programmation en CUDA Utilisation de la bibliothèque CUBLAS Optimisations de codes GPU et CPU-GPU, en CUDA Conception et mise en oeuvre d'un algorithme de K-means sur GPU

Prérequis : Cours commun de 1A : Systèmes d'Information et Programmation (1CC1000) Cours commun de 1A : Algorithmique & Complexité (1CC2000) Cours de Programmation Avancée en C++ (3MD1020) de la mention SDI à Metz Cours d'Apprentissage Automatique (3MD1040) de la mention SDI à Metz

Acquis d'apprentissage : A la fin de ce cours les élèves seront en mesure : AA1 : d'analyser l'adéquation d'une solution mathématique avec une implantation et une exécution sur GPU, AA2 : de concevoir un algorithme sur GPU, ou d'adapter un algorithme pour qu'il soit plus efficace sur GPU, AA3 : de concevoir des algorithmes hybrides CPU-GPU avec des recouvrement des transferts de données et des calculs AA4 : d'implanter des algorithmes et de mettre au point des codes sur GPU AA5 : d'analyser et de décrire synthétiquement des solutions sur GPU

Moyens : Plateforme de développement et d'exécution : serveurs de GPU du Data Center d'Enseignement du campus de Metz de CentraleSupélec. Environnements de développement CUDA de NVIDIA.

Modalités d'évaluation : Evaluation à partir des TP des parties 2 et 3 Comptes rendus des TP des parties 2 et 3 (le contenu et le nombre de pages des comptes rendus sont contraints, afin de forcer les étudiants à un effort de synthèse et de clarté) En cas d'absence non justifié à un TP la note de 0 sera appliquée, en cas d'absence justifiée la moyenne des autres TPs sera appliquée. L'examen de rattrapage sera un examen oral qui constituera 100

Compétences évaluées :
- Être opérationnel, responsable et innovant dans le monde numérique
- Savoir convaincre

CM :

1. Architecture des GPU (1.5 h)
2. Bases d'algorithmique de programmation CUDA (1.5 h)
3. Coalescence et mémoire partagée (1.5 h)
4. Optimisations avancées en CUDA (1.5 h)

TD :

1. Conception et estimation de performances d'un code CUDA (1.5 h)
2. Optimisation et accélération d'un code CUDA (1.5 h)
3. K-means sur GPU (1.5 h)

TP :

1. Produit de matrices en CUDA : implantation et expérimentation (3.0 h)

2. Produit de matrices en CUDA : optimisation et mesure de gain (3.0 h)
3. K-means sur GPU : conception, implantation et expérimentation (3.0 h)
4. K-means sur GPU : optimisations multiples (3.0 h)

Modèles parcimonieux

Responsable de cours : Laurent Duval

Total : 21.5 h (électif) **CM :** 7.0 h, **TP :** 14.0 h

3MD4020 *retour*

Description : Le cours introduit les principes de transformation des données et de méthodes d'optimisation présents au coeur de l'apprentisage automatique et de la science des données sous l'angle des notions de parcimonie et de robustesse, appliquées à la compression de données numériques (mp3, jpg) et à la représentation par des modèles prédictifs etc., en faisant largement appel à l'expérimentation algorithmique, à l'intuition et à l'histoire des sciences.

Contenu : Le cours présente un parcours en analyse de données et apprentissage via différents outils et méthodes parcimonieuses, visant à expliquer des observations par un nombre réduit de paramètres : métriques, descripteurs et transformations de données (normes, bases et trames de vecteurs, ondelettes) ; mise en oeuvre dans des algorithmes de compression de données (audio, image, vidéo, texte) ; extension aux modèles de prédictions (moments statistiques, régressions linéaires et polynomiales, modèles parcimonieux ou robustes).

Prérequis : Ce cours requiert des bases solides en algèbre linéaire et de son usage pour l'analyse des systèmes (quasi) linéaires et invariants en temps (de type filtrage) via l'analyse de Fourier (jusqu'à l'analyse harmonique), ainsi que de bonnes notions de probabilités empiriques (distributions statistiques, estimateurs). Pour les travaux pratiques, la connaisance d'un langage de scripting numérique (Matlab, Scilab, octave, Python, etc.) est requise.

Acquis d'apprentissage : Comprendre la motivation pratique et théorique d'algorithmes d'optimisation employés en apprentisage automatique et en science des données. Mettre en oeuvre les algorithmes afférents de façon adaptée en comprenant leur sens en regard du problème posé. Faire le lien entre les différentes méthodes et les mettre en oeuvre dans un flux de traitement de données.

Méthodes pédagogiques : Sur chaque thème abordé, les étudiants sont d'abord confrontés à un problème "jouet" pour lequel ils doivent mobilisent leurs connaissances, se poser des questions et implémenter des premiers algorithmes (par binôme). Dans un deuxième temps, après un échange sur cette première phase, des aspects théoriques, des preuves mathématiques et des outils algorithmiques sont présentés. Enfin, dans une troisème partie, les étudiants mettent en oeuvre ces acquis sur un problème plus complexe.

Moyens : Les cours et travaux de pratiques sont donnés par Laurent Duval (ESIEE-Paris, Université Paris-Est Marne-la-Vallée et IFP Energies nouvelles). Les cours et travaux pratiques sont entremélés, en utilisant des signaux, des images ou des données expérimentales allant de simples simulations à des données du monde réel.

Modalités d'évaluation : Le module sera évalué par un examen oral par groupe de deux ou trois étudiants, avec fourniture préalable d'un rapport, sur un sujet intégratif, conçu pour mobiliser différentes compétences et méthodes acquises dans le cours. Si le nombre d'étudiants le permet, une structure de type projet, permettant aux groupes de collaborer, sera proposée..

Compétences évaluées :
- Agir, entreprendre, innover en environnement scientifique et technologique
- Penser et agir en ingénieur éthique, responsable et intègre
- Mener un projet, une équipe

CM :

1. Introduction (1.0 h)
2. Sparse regression (1.0 h)
3. Transformations (1.0 h)
4. Approximations for compression (1.0 h)
5. Penalized sparse regression (1.0 h)
6. Complements (1.0 h)

7. Préparation aux examens (1.0 h)

TP :

1. Introduction (2.0 h)
2. Sparse regression (2.0 h)
3. Transformations (2.0 h)
4. Approximations for compression (2.0 h)
5. Penalized sparse regression (2.0 h)
6. Complements (2.0 h)
7. Préparation aux examens (2.0 h)

Apprentissage par renforcement

Responsable de cours : Hervé Frezza-Buet

Total : 23.0 h **CM :** 9.0 h, **TD :** 3.0 h, **TP :** 9.0 h

3MD4120 *retour*

Description : Le cours présente les bases théoriques de l'apprentissage par renforcement ainsi que les principes des algorithmes les plus courant. Par le biais de travaux pratiques, ces éléments seront étendus à des situations plus complexes, permettant d'introduire les algorithmes les plus récents ayant, par exemple, permis à l'ordinateur de maîtriser le jeu de Go.

Contenu : L'apprentissage par renforcement est introduit en s'appuyant sur le cadre formel des Processus Décisionnels de Markov. Après avoir montré l'existence et l'unicité d'une solution sous la forme de la fonction valeur, nous aborderons les algorithmes classiques permettant de calculer cette fonction. Nous verrons ensuite comment des méthodes approchées (approximation linéaire, estimation de monte carlo, bandits, apprentissage profond) permettent de s'attaquer à des contextes plus complexes.

Prérequis : Ce cours requiert des notions élementaires d'algèbre linéaire et de théorie des probabilités. Pour les travaux pratique, une bonne connaissance de python (numpy) est nécessaire. Le dernier TP s'appuie sur une maîtrise pratique de l'apprentissage profond avec pytorch.

Acquis d'apprentissage : Comprendre les fondements théorique de l'apprentissage par renforcement. Mettre en oeuvre ces méthodes de façon adaptée en fonction des problèmes à résoudre. Aiguiser son esprit critique.

Méthodes pédagogiques : Dans la mesure du possible (taille du groupe), les cours magistraux seront les plus interactifs possibles et auront comme objectif de présenter les notions théoriques et algorithmiques qui sous-tendent l'apprentissage par renforcement. Les travaux pratiques ont pour but de vraiment se confronter aux méthodes en implémentant et testant les algorithmes pour mieux en saisir le fonctionnement et les limites.

Moyens : Cours et travaux pratiques sont assurés par Alain DUTECH, Hervé FREZZA-BUET et Jérémy FIX. Les travaux pratiques s'appuieront sur le langage python et ses bibliothèques scientifiques.

Modalités d'évaluation : Le module sera évalué par un examen écrit, où l'idée est de tester la capacité de l'étudiant à utiliser intelligemment des méthodes, à analyser les résultats d'un algorithme, etc.

Compétences évaluées :
— Être opérationnel, responsable et innovant dans le monde numérique

---

CM :

1. Intro (1.5 h)
2. Prog. Dynamique (1.5 h)
3. Apprentissage par Renforcement (1.5 h)
4. Méthodes approchées (1.5 h)
5. Difficultés classiques (1.5 h)
6. Apprentissage par Renforcement Profond (1.5 h)

TD :

1. Modéliser une Question1/2 (1.5 h)
2. Modéliser une Question2/2 (1.5 h)

TP :

1. Probl. Académiques (3.0 h)

2. Probl. Continus (3.0 h)
3. App. Renf. Profond (3.0 h)

Traitement automatique du langage naturel

Responsable de cours : Joël Legrand

Total : 29.0 h **CM :** 9.0 h, **TP :** 18.0 h

3MD4150 *retour*

Description : Ce cours explore les fondamentaux du traitement automatique du langage naturel (TALN), couvrant des sujets tels que les word embeddings, les modèles de langue, les réseaux de neurones récurrents et récursifs, les transformers, permettant aux étudiants de maîtriser l'analyse et la génération de texte.

Contenu : Cet enseignement introduit les principales théories linguistiques permettant de modéliser le langage naturel (ex : grammaires formelles, grammaires de dépendances, ...). Il présente les différents outils de traitement automatique de langues (TAL) disponibles ainsi que modèles statistiques à la base de ceux-ci. L'accent sera notamment porté sur les méthodes d'apprentissage profond qui constituent l'état de l'art pour la plupart des tâches de TAL.

Prérequis : Maîtriser les concepts de base de l'apprentissage automatique Avoir une experience d'utilisation de librairie d'apprentissage profond (Tensorflow, pytorch, torch, ...)

Acquis d'apprentissage : À la fin de ce cours, les participants auront acquis une compréhension approfondie des concepts fondamentaux du NLP. Ils seront en mesure d'appliquer des techniques de prétraitement de texte pour nettoyer et organiser des données linguistiques, ainsi que d'utiliser des modèles de langage pré-entraînés pour diverses tâches telles que la classification de texte, la génération de texte, la traduction automatique. Les apprenants seront compétents dans l'utilisation de bibliothèques populaires de traitement du langage naturel telles que NLTK, SpaCy, Transformers.

Méthodes pédagogiques : Chaque séance comprendra une partie de cours magistral (CM) au cours duquel de nouvelles notions seront introduites, suivi d'une séance de travaux pratiques (TP) sur machine. Les TP seront des applications directes des notions vues en CM. L'ensemble du matériel pédagogique (support de CM et de TP) sera fourni aux étudiants.

Modalités d'évaluation : Examen écrit de 2h, rattrapable.

Compétences évaluées :
— Être opérationnel, responsable et innovant dans le monde numérique

---

CM :

1. Word representations (1.0 h)
2. Language models (1.0 h)
3. Sequence labeling (1.0 h)
4. Sentence classification (1.0 h)
5. Syntactic analysis : constituency parsing (1.0 h)
6. Syntactic analysis and RNN (1.0 h)
7. Machine translation (1.0 h)
8. Machine translations with Seq2seq RNN and attention mechanisms (1.0 h)
9. Le modèle transformer (1.0 h)

TP :

1. Word representations (2.0 h)
2. Word embeddings (2.0 h)
3. Sequence labeling (2.0 h)
4. LSTM (2.0 h)
5. RNN language model (2.0 h)

6. Sentiment analysis with RNN (2.0 h)
7. Machine translation (2.0 h)
8. Machine translations with Seq2seq RNN and attention mechanisms (2.0 h)
9. Le modèle transformer (2.0 h)

Apprentissage statistique

Responsable de cours : Paul Fraux

Total : 23.0 h **CM :** 10.5 h, **TD :** 10.5 h

3MD4140 *retour*

Description : L'apprentissage supervisé a pour objectif de proposer des méthodes qui, à partir d'une base d'exemples, permettent de prendre une décision portant sur un paramètre à partir d'observations, la décision devant être la meilleure possible en moyenne. Par exemple, classifier des images suivant leur contenu, c'est-à-dire décider si une image représente un chat, un chien, ou autre chose. Nous présenterons formellement le problème et étudierons les garanties de généralisation des algorithmes d'apprentissage supervisé, c'est-à-dire la qualité de prédiction de la sortie associée à une entrée non présente dans la base d'entraînement. Pour atteindre cet objectif, nous introduirons les concepts d'espace d'hypothèses ayant la capacité d'apprentissage PAC (probablement approximativement correcte), de dimension Vapnik-Chervonenkis d'un espace d'hypothèses. Nous énoncerons et démontrerons deux théorèmes fondamentaux de la théorie de l'apprentissage supervisé donnant une borne inférieure et une borne supérieure du risque réel au problème de classification binaire.

Contenu : Formalisation du problème de l'apprentissage supervisé Capacité d'apprentissage PAC et convergence uniforme Le dilemme biais-complexité La dimension VC (Vapnik-Chervonenkis) d'un espace d'hypothèse Deux théorèmes fondamentaux de l'apprentissage supervisé

Prérequis : cours de Probabilités de 1A (CIP-EDP, 1SL1000) cours de Statistique et apprentissage de ST4 (1CC5000)

Acquis d'apprentissage : A l'issue de ce cours, les élèves devront être en mesure - de comprendre et s'approprier des éléments de la théorie de l'apprentissage supervisé ; - de comprendre et s'approprier le dilemme biais-complexité d'un espace d'hypothèses ; - de comprendre et s'approprier les bornes bayésiennes PAC de l'apprentissage supervisé (en particulier celles du problème de classification binaire).

Méthodes pédagogiques : 10,5h de cours magistraux + 10,5h de travaux dirigés + examen écrit de 2h

Moyens : Les travaux dirigés, constitués d'exercices, permettront d'utiliser les concepts vus en cours.

Modalités d'évaluation : Examen écrit de 2h avec documents

Compétences évaluées :
— Analyser, concevoir et réaliser des systèmes complexes

CM :

1. Modèle formel de l'apprentissage statistique supervisé (1.5 h)
2. Capacité d'apprentissage PAC (1.5 h)
3. Dilemme biais-complexité (1.5 h)
4. No free lunch theorem (1.5 h)
5. Dimension VC (1.5 h)
6. Théorèmes fondamentaux de l'apprentissage PAC (1.5 h)
7. Théorèmes fondamentaux de l'apprentissage PAC (1.5 h)

TD :

1. Rappels et compléments mathématiques (1.5 h)
2. Rappels et compléments mathématiques (1.5 h)
3. Prédicteurs linéaires (1.5 h)
4. Prédicteurs linéaires (1.5 h)
5. Théorèmes fondamentaux de l'apprentissage PAC (1.5 h)

6. Théorèmes fondamentaux de l'apprentissage PAC (1.5 h)
7. Théorèmes fondamentaux de l'apprentissage PAC (1.5 h)

Modèles de calcul du Big Data

Responsable de cours : Stéphane Vialle

Total : 21.0 h **CM :** 10.5 h, **TD :** 1.5 h, **TP :** 9.0 h

3MD4130 *retour*

Description : Ce cours a pour objectif d'apprendre aux élèves à développer des applications performantes d'analyse de données en environnement Spark sur des plates-formes distribuées (clusters et Clouds). Des mécanismes de systèmes de fichiers distribués comme HDFS seront étudiés, ainsi que le modèle de programmation et l'algorithmique du "map-reduce étendu" de Spark au dessus des Spark "RDD", puis des modèles de programmation de plus haut niveau au dessus de Spark "Data Frames", et enfin des modèles de programmation sur Cloud. Des critères et métriques de passage à l'échelle seront également étudiés. Tout au long du cours des mises en oeuvres auront lieu sur des clusters et dans un Cloud, et les solutions développées seront évaluées par les performances obtenues sur les cas-tests, et par leur aptitude à passer à l'échelle.

Contenu : Emergence des technologies Big Data : motivations, besoins industriels, principaux acteurs. Pile logicielle d'Hadoop, architecture et fonctionnement de son système de fichier distribué (HDFS) Architecture et mécanisme de déploiement de calculs distribués de Spark Modèle de programmation par "RDD" et algorithmique du "map-reduce" étendu de Spark Modèle de programmation de Spark par "Data Frames" appliqué à l'analyse de graphes (GraphX) Architecture et environnement d'analyse de données sur Cloud Expérimentations et mesures de performances Critères et métriques de performances

Acquis d'apprentissage : A l'issue de ce cours, les élèves seront en mesure :

AA1 : de concevoir et d'implanter des algorithmes "map-reduce étendu" en Spark, efficaces sur des plates-formes distribuées, et passant à l'échelle. AA2 : d'analyser les capacités de "passage à l'échelle" d'une application AA3 : d'utiliser un cluster ou un cloud pour réaliser des analyses de données distribuées à large échelle. AA4 : de présenter synthétiquement une solution d'analyse de données conçue en "map-reduce"

Méthodes pédagogiques : Ce cours enchaîne 3 parties relatives à des modèles de calculs du "Big Data" : la première sur clusters de PC, la seconde dans les Cloud, et la troisième qui évalue des solutions de "passage à l'échelle".

Plan du cours en 4 parties :

Partie 1 : Architecture et développement en Spark RDD sur HDFS et clusters de PC.

Partie 2 : Critères et métriques pour la performance et le passage à l'échelle.

Partie 3 : Calcul et analyse de données large échelle sur Cloud.

Partie 4 : Développement en Spark Data Frames sur HDFS et clusters de PC.

Moyens : Equipe enseignante : Stéphane Vialle et Gianluca Quercini (CentraleSupelec), Wilfried Kirschenmann (ANEO) Plateforme de développement et d'exécution : clusters de calcul du Data Center d'Enseignement (DCE) du campus de Metz de CentraleSupélec accès à un Cloud professionnel Environnements de développement : Spark + HDFS sur le DCE autre environnement sur le Cloud

Modalités d'évaluation : Evaluation à partir des TP :

Les comptes rendus des TP seront notés (le contenu et le nombre de pages des comptes rendus sont contraints, afin de forcer les étudiants à un effort de synthèse et de clarté) En cas d'absence non justifié à un TP la note de 0 sera appliquée, en cas d'absence justifiée le TP n'interviendra pas dans la note finale. L'examen de rattrapage sera un examen oral, qui constituera 100

Compétences évaluées :
- Être opérationnel, responsable et innovant dans le monde numérique
- Savoir convaincre

CM :

1. Emergence du Big Data et technologie HDFS d'Hadoop (1.5 h)
2. Technologie Spark-RDD et programmation Map-Reduce (1.5 h)
3. Optimisation et déploiement de codes Map-Reduce (1.5 h)
4. Métriques de passage à l'échelle et architecture des Data Lakes (1.5 h)
5. Spark Data-Frames et Spark SQL (1.5 h)
6. Problématiques du cloud (1.5 h)
7. Environnement de développement sur cloud (1.5 h)

TD :

1. Algorithmique Map-Reduce en Spark (1.5 h)

TP :

1. Map-Reduce sur cluster Spark-HDFS : programmation et performances (3.0 h)
2. Analyse de données en Spark Data-Frame et Spark SQL sur cluster Spark-HDFS (3.0 h)
3. Développement et déploiement d'une analyse de données extensible sur Cloud (3.0 h)

Perspectives en Apprentissage et Intelligence Artificielle

Responsable de cours : Joël Legrand

Total : 23.5 h **CM :** 8.5 h, **TP :** 15.0 h

3MD4110 *retour*

Description : Ce module propose une ouverture vers des applications concrètes de l'apprentissage automatique à travers des interventions de chercheurs et d'industriels. Chaque séance met en lumière un domaine d'application spécifique (santé, finance, énergie, robotique, etc.), illustrant comment des techniques classiques ou avancées sont mobilisées pour répondre à des problématiques réelles. Ces interventions complètent les enseignements académiques en exposant les étudiants à des approches parfois plus poussées, favorisant ainsi une compréhension élargie des défis et des pratiques actuelles en milieu professionnel ou en recherche.

Prérequis : Ce module requière une bonne maîtrise des concepts d'apprentissage automatique abordés dans les différents cours de la mention SDI-Metz (Deep learning, Natural language processing, Machine Learning, ...).

Acquis d'apprentissage : À l'issue de ce module, les étudiants auront acquis une vision élargie des applications concrètes de l'apprentissage automatique dans différents secteurs. Ils seront capables d'analyser des problématiques réelles sous l'angle de l'IA, de comprendre les choix méthodologiques faits par des experts, et d'identifier les contraintes spécifiques liées à l'implémentation de solutions en contexte industriel ou de recherche. Ce module développera également leur capacité à dialoguer avec des professionnels du domaine et à se projeter dans des projets interdisciplinaires mobilisant l'apprentissage automatique.

Moyens : Chaque module (5 au total) est assuré par un industriel ou un chercheur. Il est composé d'un cours magistral de 1h30 suivi d'une session pratique de 3h.

Modalités d'évaluation : À la fin de chaque session pratique, un rendu sera exigé par l'intervenant. Un seul rendu sera ensuite sélectionné aléatoirement pour être évalué par l'intervenant en question. La note attribuée servira d'évaluation pour l'ensemble du module.

Compétences évaluées :
— Être opérationnel, responsable et innovant dans le monde numérique

---

CM :

1. Introduction (1.0 h)
2. Conférence 1 (1.5 h)
3. Conférence 2 (1.5 h)
4. Conférence 3 (1.5 h)
5. Conférence 4 (1.5 h)
6. Conférence 5 (1.5 h)

TP :

1. TP 1 (3.0 h)
2. TP 2 (3.0 h)
3. TP 3 (3.0 h)
4. TP 4 (3.0 h)
5. TP 5 (3.0 h)