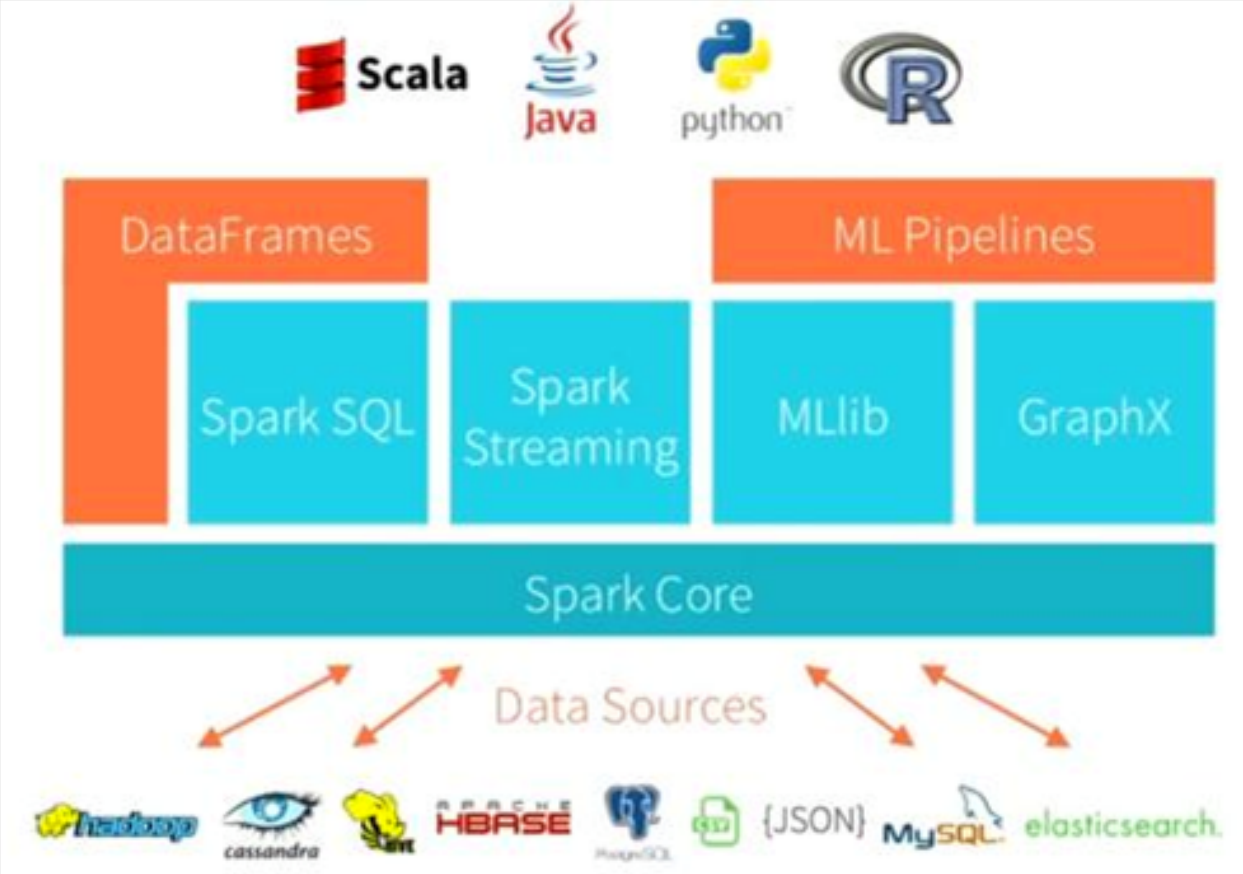
Technologies de l'IA
Marketing & *Spark-MLlib*

Philippe Besse & Brendan Guillouet

Université de Toulouse
INSA – Dpt GMM
Institut de Mathématiques – ESP
UMR CNRS 5219



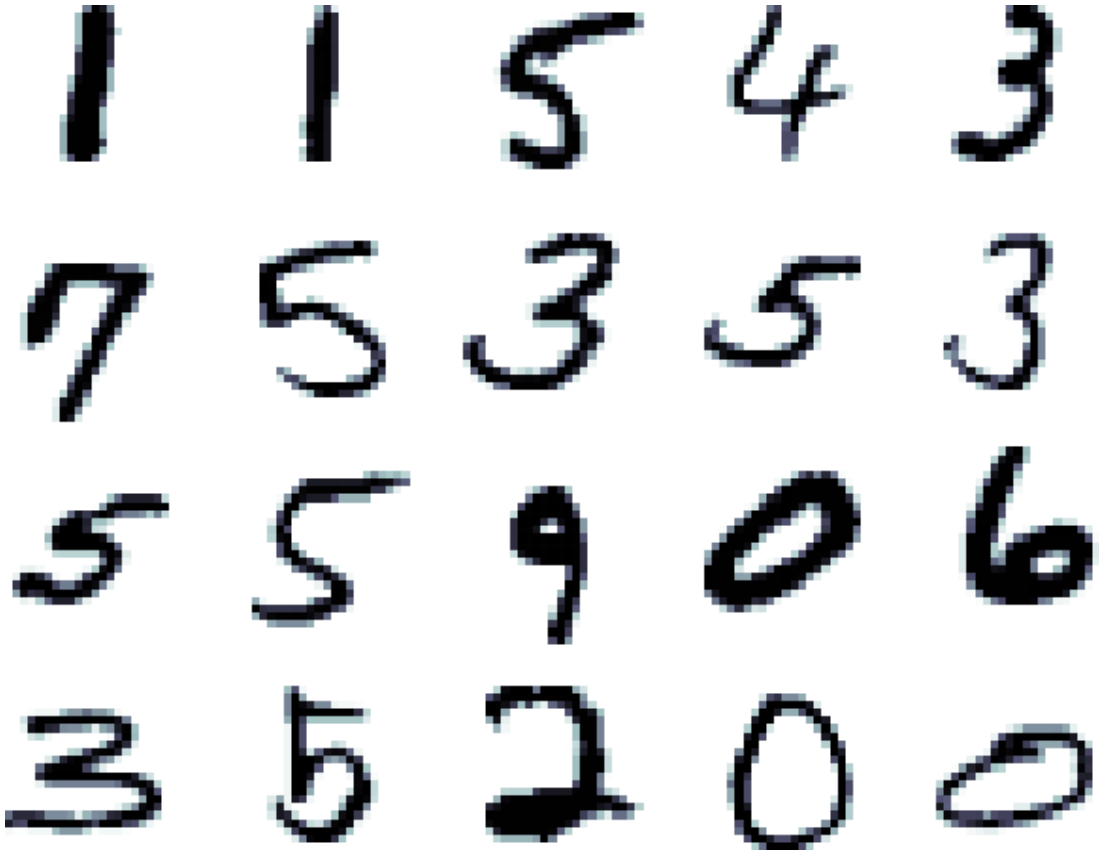
La technologie Spark et son écosystème

Librairie *SparkML vs. MLlib*

- Spark 2.4
- Peu de méthodes mais passage à l'échelle "Volume"
- MLlib : *Resilient Distributed Dataset (RDD)*
- SparkML : *DataFrame*, *pipeline* vers *MLlib*
- Fonctions
  - Statistique descriptive, tests
  - k-*means*, GMM
  - SVD donc ACP
  - Non negative Matrix Factorisation (ALS)
  - Régression linéaire et logistique (l_1 et l_2)
  - SVM linéaires, Classifieur Bayésien Naïf
  - Arbre, Forêt Aléatoire, Boosting (GBM)

Apprentissage avec *Spark MLlib*

- Données massives : n et p très grands
- Données distribuées :*Hadoop*
- Après préparation des données massives avec *Spark*
- **Question** :
  - Apprendre sur tout : *Spark MLlib*
  - ou échantillonner : *Python Scikit-learn*
- **Exemples**
  - *Random Forest* sur MNIST
  - *Non negative Matrix Factorisation (NMF)*
    Movie Lens : recommandation de films
  - Régression logistique
    Catégorisation de textes de Cdiscount



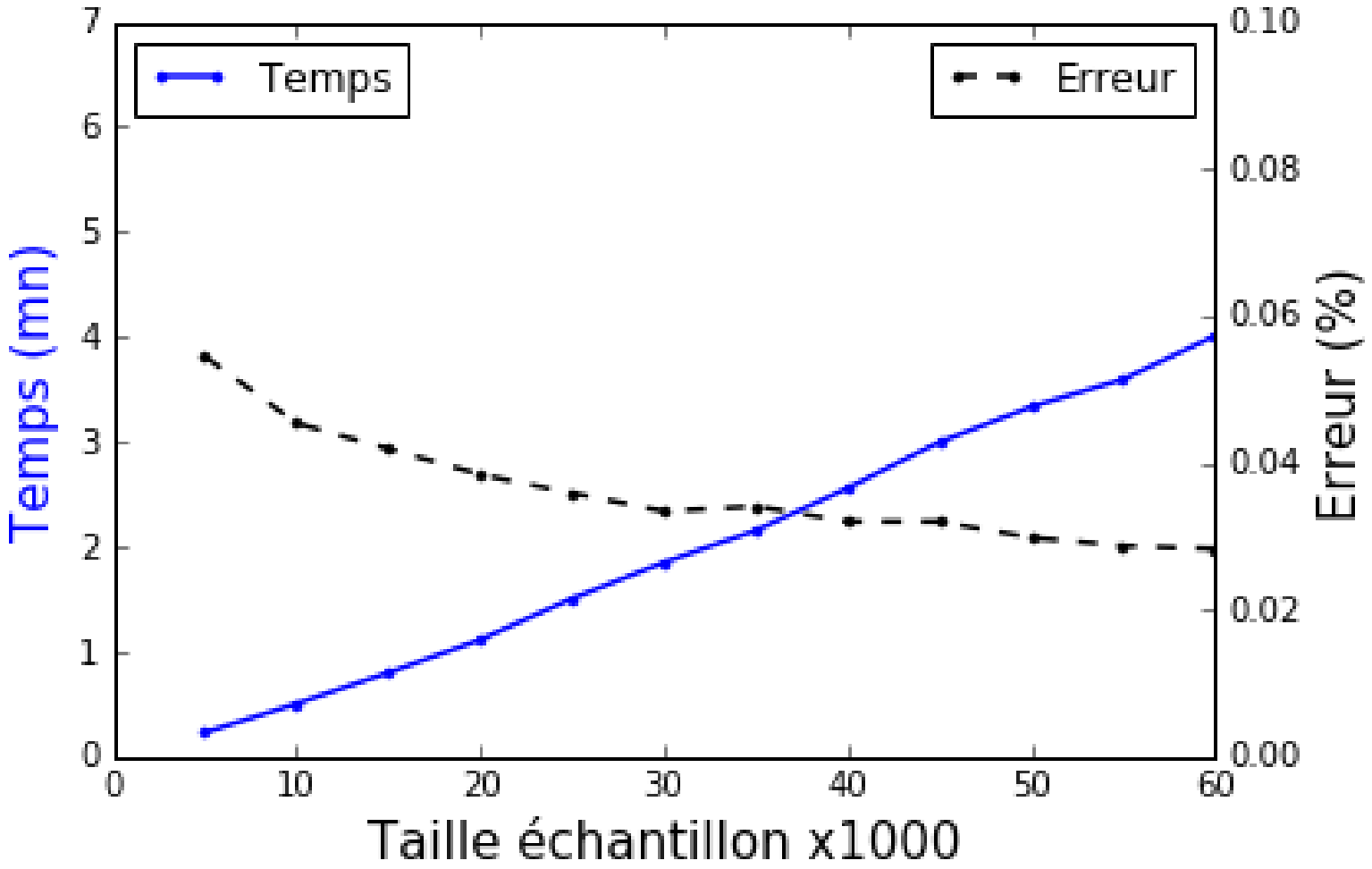
MNIST : quelques exemples d'images de caractères

MNIST : État de l'art

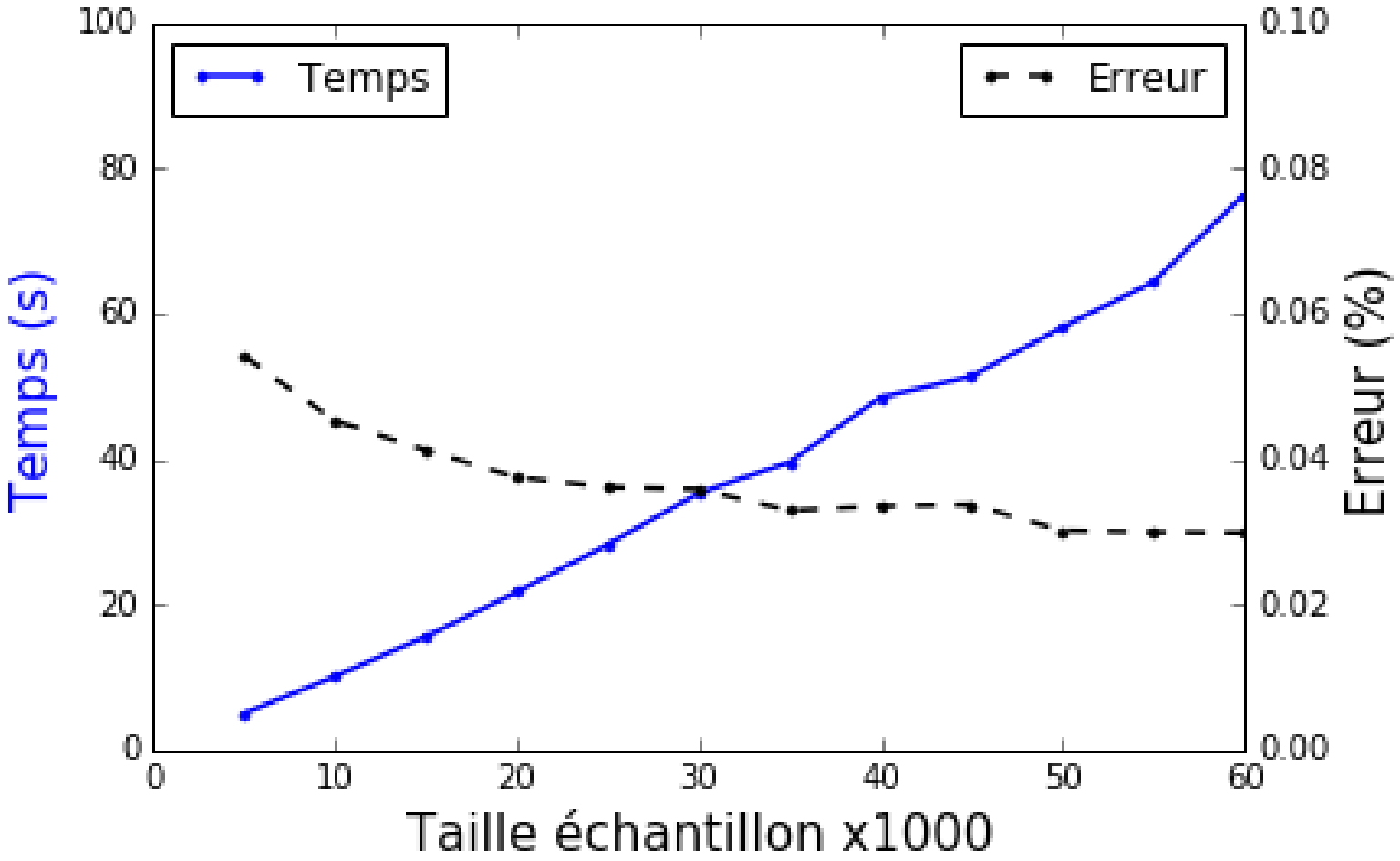
- Site de Yann le Cun
- 60 000 caractères, $28 \times 28 = 784$ pixels
- Test : 10 000 images
- Méthodes classiques (k-nn, RF)
- Prétraitement de normalisation des images
- Distance spécifique (tangeantielle) avec propriétés d'invariance
- Ondelettes et *scattering* (Stéphane Mallat)
- Apprentissage Profond : *TensorFlow, Lasagne, Keras*
- ...
- Comparer l'usage des méthodes classiques

Implémentations de Random Forest

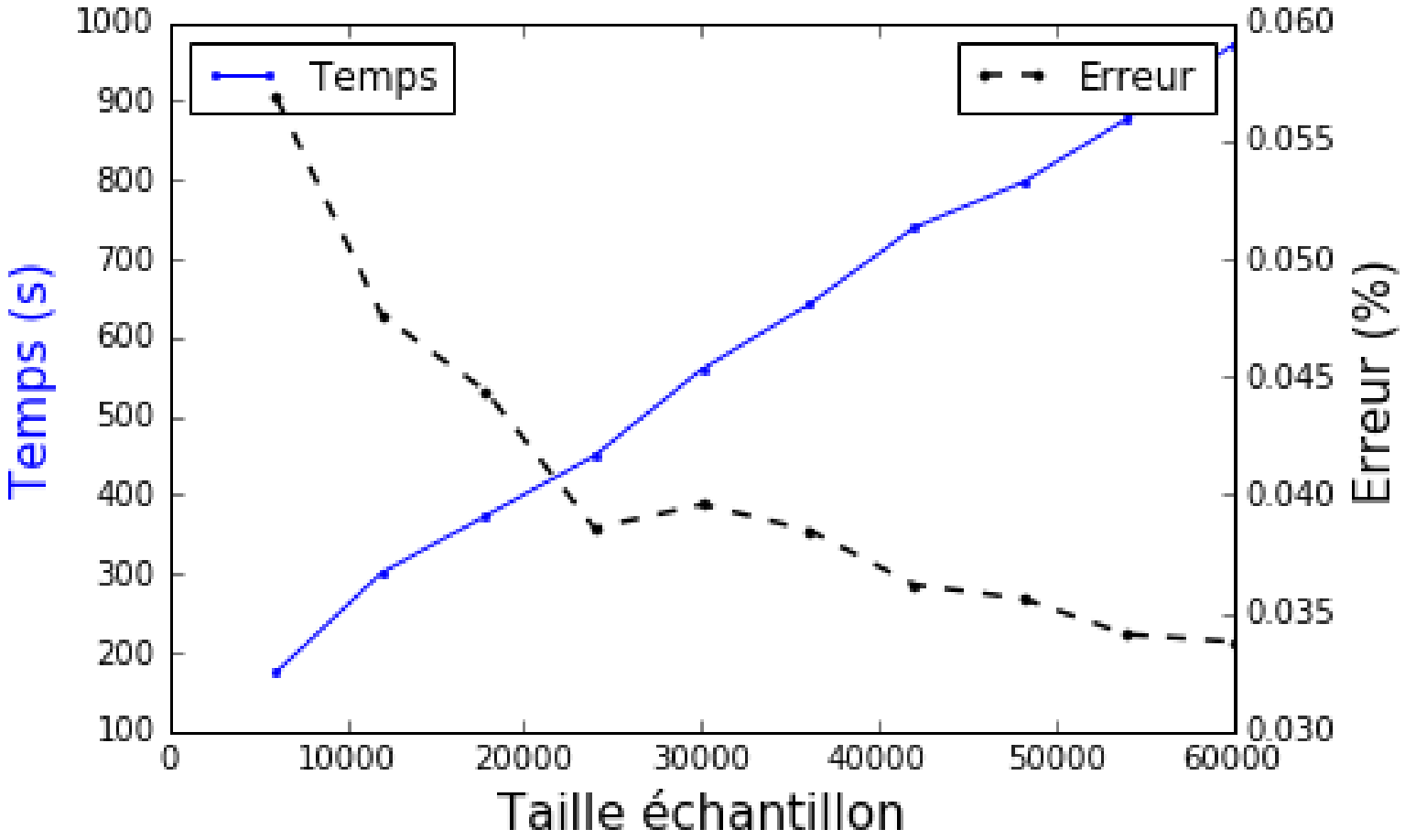
- R : `randomForest, ranger`
- Python : `Scikit-learn`
- MLlib : arbres du projet Google PLANET
- Problèmes : mémoire et temps d'exécution
  - `maxBins = 32`
  - `maxDepth` : élagage d'un arbre
- *Bootstrap* ou tirages sans remise ?



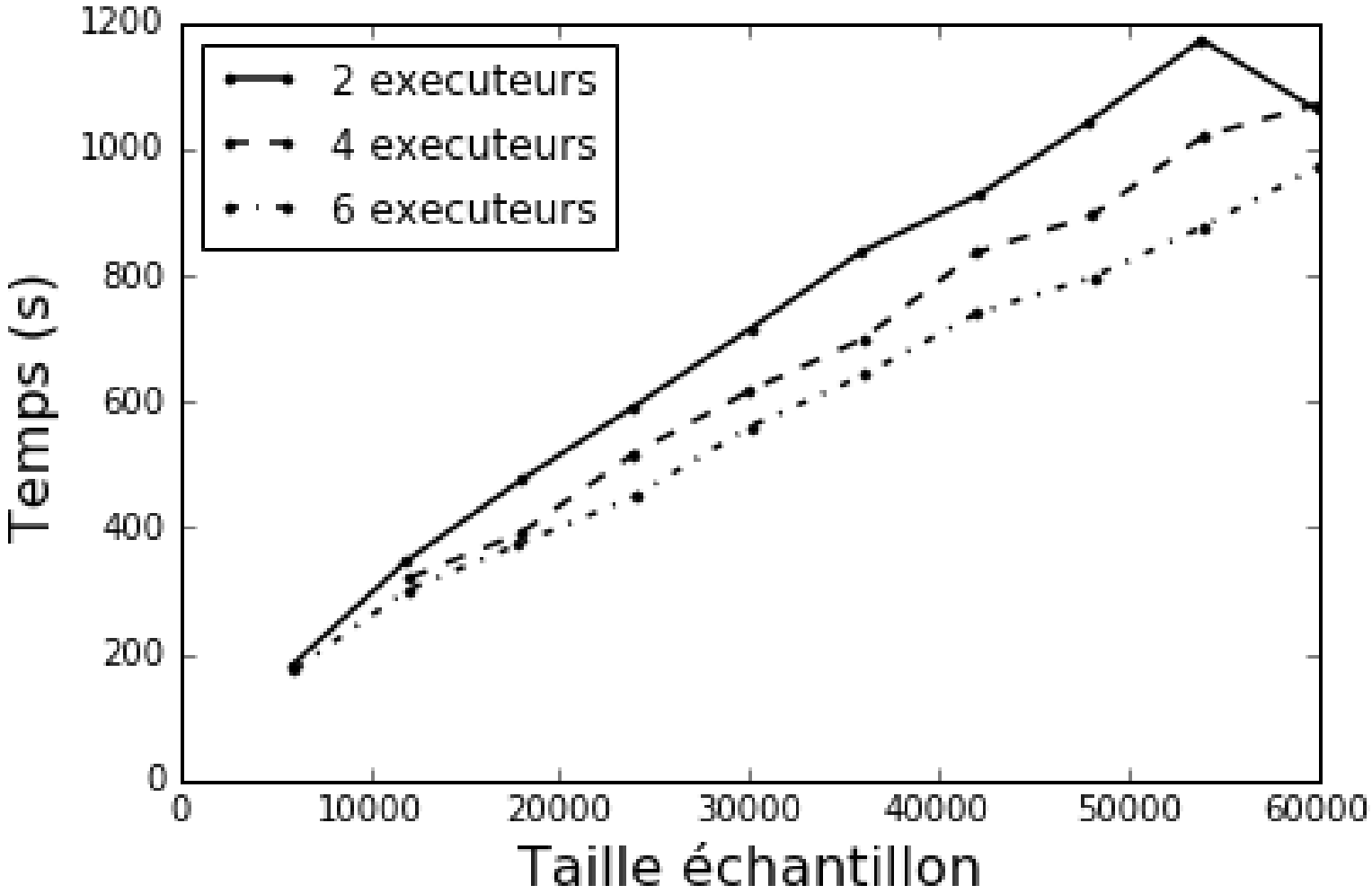
MNIST : forêts aléatoires avec R (`ranger` ***pas*** `randomForest`*)*



MNIST : forêts aléatoires avec Python (Scikit-learn)



MNIST : forêts aléatoires avec Spark (MLlib) ; `maxdepth=15`



MNIST : Forêts aléatoires (Spark) avec 2, 4 ou 6 exécuteurs

MNIST : discussion

- R `ranger` (sauf Windows) ou Python `Scikit-learn`
- Spark : problèmes de mémoire limitent `maxDpeth`, nb arbres
- Spark : *scalability* pas vérifiée
- Plateau de l'erreur fonction de la taille de l'apprentissage
- Architecture intégrée préférable à distribuée
- Conclusion : *Scikit-learn* plutôt que *MLlib*

E-commerce et Systèmes de Recommandation

- Gestion de la relation client : score d'appétence
- *Google double click*, *Criteo*, enchères
- Commerce en ligne et filtrage de produits (Cf. Vignette)
  - Filtrage adaptatif : algorithme de bandit
  - Filtrage collaboratif : concours *Netflix*
    - Basé sur les seules informations client $\times$ produit
    - Modèle de voisinage
    - Facteurs latents
    - Méthodes mixtes ou hybrides

Problèmes délicats

- Évaluation d'une recommandation :
  Nb clics de ventes, *AB testing*, RMSE
- Initialisation : *cold start*

Filtrage collaboratif par voisinage

- Grande matrice creuse clients $\times$ produits
  - Nombre ou présence de ventes
  - Appréciation ou note de 1 à 5
- Basé sur le voisinage client ou produit
  - Définir une similarité (corrélation linéaire ou rang)
  - Trouver un sous-ensemble S_i de proches de i
  - Prévoir une note ou un achat
  - Combinaison linéaire des notes de S_i

Filtrage collaboratif par facteurs latents

- Matrices clients $\times$ produits très creuses
- Nombre d'achats ou de clics *vs.* appréciation ou note
- Valeur nulle *vs.* valeur manquante
- Modèle par factorisation *vs.* Complétion
  - Candes et Tao (2010)
  - $\min_{\mathbf{M}} \left(||P_\Omega(\mathbf{X} - \mathbf{M})||_2^2 + \lambda ||\mathbf{M}||_* \right)$
  - $P_\Omega(\mathbf{X})$: "restriction" de la matrice $\mathbf{X}$ à l'ensemble des valeurs connues

Complétion de matrice

- *Netflix* ou *MovieLens* : problème de complétion
- Sujet à la bibliographie explosive
- Deux méthodes facilement accessibles (R et Spark)
  - Librairie R `softImpute` : Algorithme hybride associant SVDs seuillées et moindres carrés alternés (Hastie et al. 2015)

$$\min_{\mathbf{A_{n\times r}},\mathbf{B_{p\times r}}} ||P_\Omega(\mathbf{X}-\mathbf{AB}^T)||_2^2 + \lambda\left(||\mathbf{A}||_2^2 + ||\mathbf{B}||_2^2\right)$$

  - Librairie Spark MLlib : Complétion par *Non negative Matrix Factorisation* (NMF)

Non negative Matrix Factorisation

$$\min_{\mathbf{W},\mathbf{H}\geq 0} \left[l(\mathbf{X}, \mathbf{WH}) + P(\mathbf{W}, \mathbf{H}) \right]$$

- Similaire à la SVD mais avec contrainte de non négativité
- l : moindres carrés ou Kulback Leibler
- P : régularisation
- Nombreux algorithmes dont ALS
- Convergence locale
- Nombreuses initialisations disponibles
- Optimiser le rang de $\mathbf{W}$ et $\mathbf{H}$
- Optimiser le coefficient de régularisation
- *MLlib* : deux options factorisation ou complétion : $P_\Omega(\mathbf{X})$

Données MovieLens

`100k` 100 000 évaluations de 1000 utilisateurs de 1700 films.

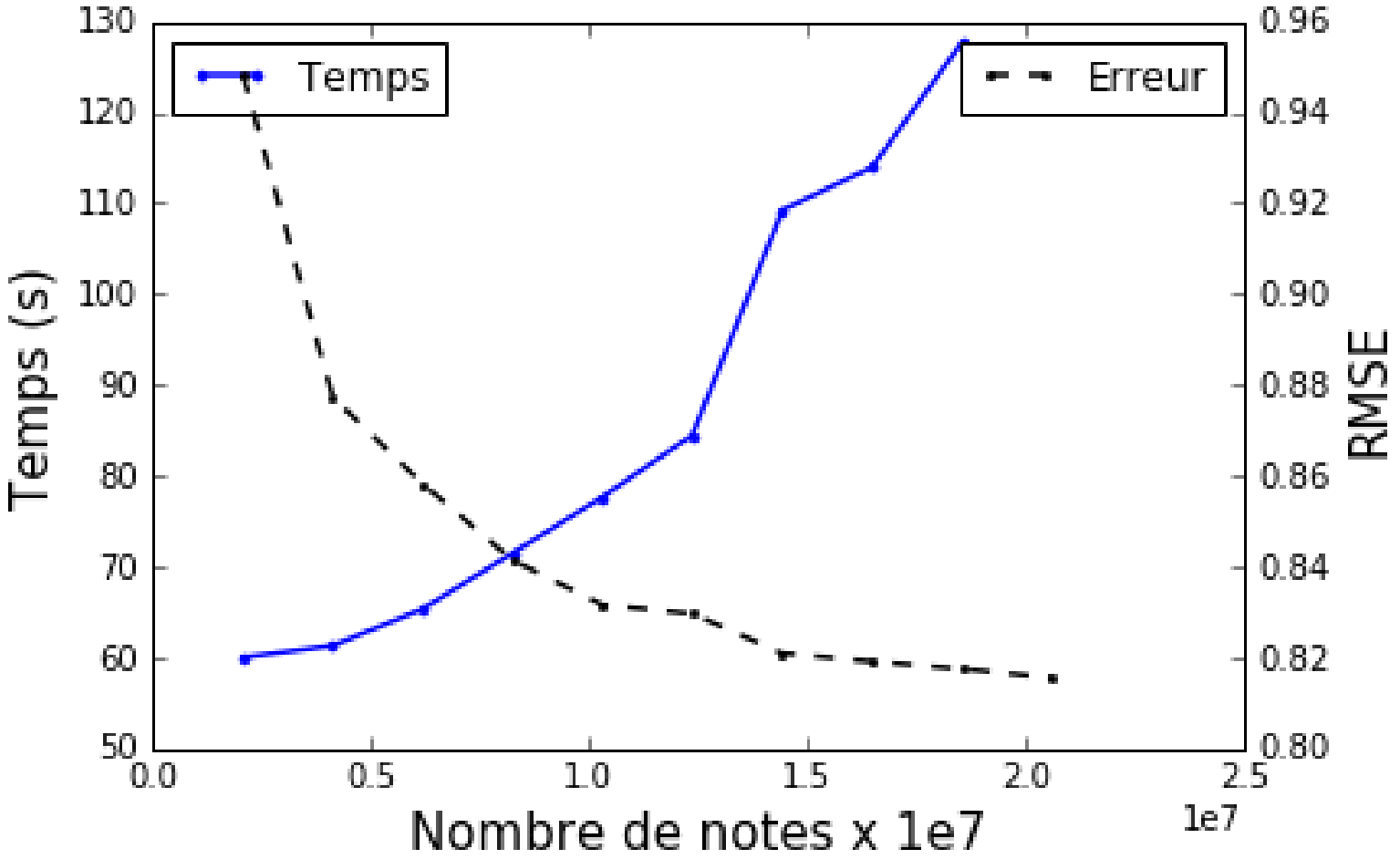
`1M` Un million d'évaluations par 6000 utilisateurs sur 4000 films.

`10M` Dix millions d'évaluations par 72 000 utilisateurs sur 10 000 films.

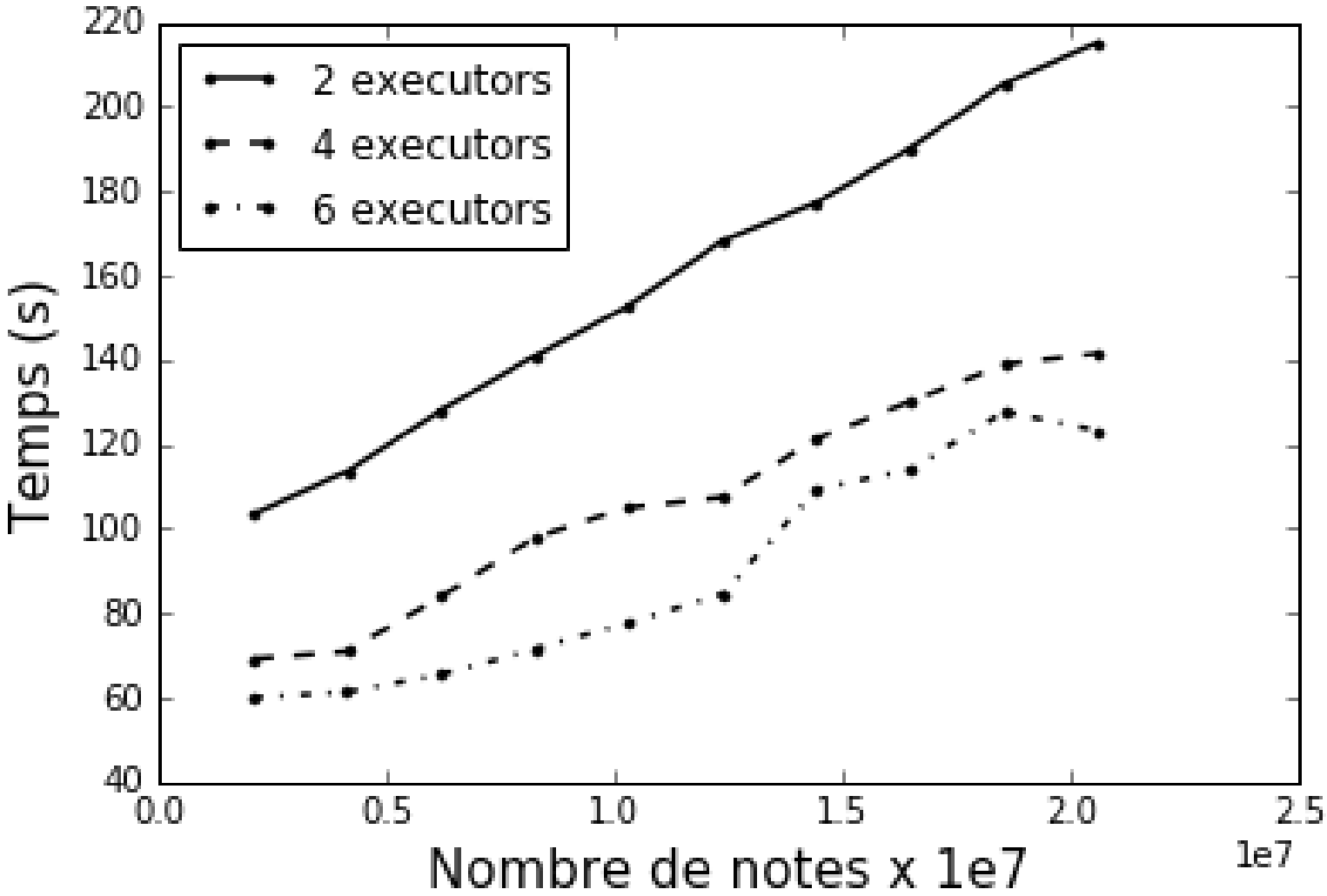
`20M` Vingt deux millions d'évaluations par 138 000 utilisateurs sur 27 000 films.

Rang Max	λ	Temps	RMSE
4	1	5.6	1.07
10	10	12.6	1.020
10	20	12.2	1.033
15	10	19.4	1.016
20	1	26.9	1.020
20	10	26.1	1.016
20	15	24.4	1.018
20	20	27.0	1.016
30	20	40.1	1.020

MovieLens : Optimisation du rang et de la régularisation de softImpute



MovieLens : complétion par NMF (MLlib)



MovieLens : complétion par NMF (MLlib)

MovieLens : discussion

- NMF mais pas de complétion dans `Scikit-learn`
- Moins bon RMSE de *softImpute* : pas de contrainte ?
- MLlib : pas tout à fait *scalable*
- Architecture distribuée adaptée aux matrices creuses

Conclusion

- Comparaison entre architectures distribuée *vs.* intégrée
- Problème de maturation des technologies
- Trois étapes :
  - *Data munging*, *streaming* : Spark, SparkSQL
  - Apprentissage : grosses data et gros modèles
- R *vs.* Python Scikit-learn *vs.* SparkSQL, SparkML
- Nettoyage et Apprentissage en ligne ?
- But ? Publication, Concours, Industriel
- Cdiscount, Critéo, Deepky, Tinyclues, Hupi (ppml)...
- Ne pas oublier : fiabilité, représentativité des données