Régression sur variables entachées d'erreurs à l'aide de réseaux de neurones bayésiens : présentation d'une approche motivée par la datation carbone 14

Destin Ashuza Cirumanga [1], Anne Philippe [2] & Guillaume Guérin [3]

[1] *CNRS & Laboratoire de Mathématiques Jean Leray, Nantes Université, France.*
destin.ashuzacirumanga@univ-nantes.fr
[2] *Laboratoire de Mathématiques Jean Leray, Nantes Université, France.*
anne.philippe@univ-nantes.fr
[3] *CNRS & Géosciences Rennes, Université Rennes 1, France.*
guillaume.guerin@univ-rennes1.fr

Résumé. L'estimation d'une fonction de régression par un modèle paramétrique en présence d'une variable explicative bruitée conduit à une estimation biaisée et non consistante si on ne prend pas en compte l'incertitude en entrée du modèle. Des solutions existent pour la prise en compte de cette incertitude dans le cadre des modèles linéaires et non linéaires classiques. Cependant, elles ne sont pas adaptées aux réseaux de neurones. En utilisant l'approximation variationnelle gaussienne combinée à une modélisation jointe de la variable explicative entachée d'erreur et de la variable dépendante, nous proposons une approche permettant de prendre en compte l'incertitude en entrée du réseau tout en préservant une simplicité d'entraînement similaire à celle du cas standard d'un apprentissage sur des entrées non bruitées. De plus cet apprentissage peut se faire en mode distribué. Nous validons notre approche sur des données simulées ayant des caractéristiques similaires aux données réelles de datation par le carbone 14 en archéologie qui ont motivé ce travail.

Mots-clés. Réseaux de neurones, statistique bayésienne, inférence variationnelle, variables entachées d'erreurs.

Abstract. Estimation of a regression function by a parametric model in the presence of a noisy input variable leads to a biased and inconsistent estimate if the uncertainty in the model input is not taken into account. Solutions exist for taking this uncertainty into account in the context of classical linear and non-linear models. However, they are not suitable for neural networks. By using the Gaussian variational approximation combined with joint modeling of the error-prone input variable and the dependent variable, we propose an approach that takes into account the uncertainty in the network input while preserving a training simplicity similar to that of the standard case of learning on noiseless inputs. What's more, this training can be carried out in distributed mode. We validate our approach on simulated data with characteristics similar to the real archaeological carbon-14 dating data that motivated this work.

Keywords. Neural networks, Bayesian statistics, variational inference, errors-in-variables models.

1 Introduction à la problématique des variables explicatives entachées d'erreurs en régression

En apprentissage statistique, quand on veut estimer un modèle de régression, il est fréquent de considérer que les variables explicatives sont déterministes et que seule la variable à expliquer est bruitée. Cependant, dans certaines situations, les variables explicatives peuvent être entachées d'erreurs. La problématique de l'estimation de la courbe de calibration des âges carbone 14 traitée dans Heaton et al. (2020) en est un exemple. En effet, la calibration des âges carbone 14 nécessite l'estimation d'une fonction de régression qui prend en entrée la date (âge) d'un objet donné et donne en sortie l'âge carbone 14 correspondant. Mais les dates utilisées sont entachées d'erreurs pour certaines périodes de temps.

La présence d'incertitude dans les variables explicatives conduit aux modèles dits de régression sur variables entachées d'erreurs. Ces modèles ont été largement étudiés dans la littérature dans le cadre de la régression linéaire. Un aperçu de l'état de l'art sur la question peut être trouvé dans Carroll et al. (2006). Un constat est que la non prise en compte de l'erreur dans une variable explicative conduit à une estimation biaisée des paramètres du modèle. Ce phénomène est connu sous le nom de biais d'atténuation.

De nombreuses méthodes pour la prise en compte de l'erreur dans une variable explicative ont été proposées pour les modèles linéaires et les modèles linéaires généralisés. C'est le cas par exemple des méthodes de régression - calibration, simulation - extrapolation, variable instrumentale ou certaines fonctions de score. Comme souligné dans Carroll et al. (2006), ces méthodes ont des performances faibles pour des modèles fortement non linéaires. Pour les modèles plus complexes ou fortement non linéaires, des alternatives existent parmi lesquelles des approches basées sur les méthodes de vraisemblance ou les modèles bayésiens hiérarchiques. Dans Heaton et al. (2020) par exemple, une approche bayésienne hiérarchique couplée aux splines est utilisée pour estimer la courbe de calibration des âges carbone 14. Notre objectif est de proposer une estimation de cette courbe par des réseaux de neurones bayésiens.

La problématique de la prise en compte de l'incertitude dans une variable explicative pour les réseaux de neurones a été peu étudiée. Martin and Elster (2023) propose une approche basée sur l'algorithme de Monte Carlo dropout. Nous proposons dans ce papier une alternative basée sur l'approximation variationnelle gaussienne. La suite de ce papier se subdivise en trois points : dans la section 2 nous formalisons le problème et dérivons la fonction de perte à optimiser pour estimer le modèle ; sur base de cette fonction de perte, dans la section 3 nous proposons deux stratégies d'apprentissage permettant de prendre en compte l'incertitude en entrée du réseau tout en conservant une grande simplicité dans l'entraînement du réseau ; et enfin dans la section 4 nous présentons les résultats obtenus par notre approche sur des données simulées.

2 Modélisation et estimation de la fonction de régression

Nous disposons des données d'apprentissage $\mathcal{D} = \{(t_i, \sigma_i, M_i, s_i), i = 1, \cdots, n\}$ avec
- d_i : la vraie valeur de la variable explicative inconnue,
- t_i : la variable explicative d_i entachée d'erreur,
- σ_i : l'écart-type de l'erreur associée à t_i,
- M_i : la variable à expliquer bruitée, et
- s_i : l'écart-type du bruit associé à M_i.

Les variables observées t_i et M_i sont distribuées suivant des lois normales de telle sorte que le modèle qui en découle s'écrive comme suit :

$$\begin{cases} M_i = m_i + s_i \cdot \varepsilon_i \\ m_i = g(d_i, \theta) \\ t_i = d_i + \sigma_i \cdot u_i \end{cases} \tag{1}$$

avec $\varepsilon_i \sim \mathcal{N}(0,1)$, $u_i \sim \mathcal{N}(0,1)$, g la fonction de régression que l'on souhaite estimer par un réseau de neurones bayésien de paramètres (les poids et biais du réseau) θ dont la loi *a priori* $\pi(\theta)$ est une gaussienne multivariée de composantes toutes indépendantes, centrées et réduites. En l'absence des connaissances particulières, nous choisissons une loi *a priori* non informative pour d_i : la loi de Laplace i.e $\pi(d_i) \propto \mathbb{1}_{\mathbb{R}}(d_i)$.

Plus précisément, si le réseau de neurones est composé de L couches, la fonction de régression g s'écrit alors comme suit :

$$g(d_i, \theta) = \Phi_L \circ \Phi_{L-1} \circ \cdots \circ \Phi_1(X_0) \tag{2}$$

avec $\theta = \{(\omega_j, b_j)_{1 \leq j \leq L}\}$, l'entrée $X_0 = d_i$ et pour toute couche $j \in \{1, \cdots, L\}$, le vecteur des sorties X_j est donné par $X_j = \Phi_j(X_{j-1}) = \phi_j(\omega_j X_{j-1} + b_j)$, ω_j et b_j étant respectivement la matrice des poids et le vecteur des biais de neurones de cette couche, et ϕ_j sa fonction d'activation. Pour tout $x \in \mathbb{R}$, on a $\phi_L(x) = x$ et $\phi_j(x) = \max(0, x)$ pour chaque $j \in \{1, \cdots, L-1\}$ (la fonction $ReLU$). Dans la notation $\phi_j(\omega_j X_{j-1} + b_j)$, la fonction d'activation ϕ_j est évaluée composante par composante. Ces spécifications relatives à la fonction g définissent un perceptron multicouche pour notre problème. Plus de détails sur les perceptrons et d'autres architectures de réseaux de neurones peuvent être trouvés dans James et al. (2021).

Pour spécifier intégralement notre modèle bayésien, il reste à déterminer sa vraisemblance. Comme présentées dans Carroll et al. (2006), les méthodes de vraisemblance pour les modèles de régression sur variables entachées d'erreurs sont basées sur le calcul de la densité jointe de la variable à expliquer et de la variable explicative entachée d'erreur. Pour tout $i \in \{1, \cdots, n\}$, la densité $p(M_i, t_i | \theta, s_i, \sigma_i)$ de (M_i, t_i) s'écrit :

$$\begin{aligned} p(M_i, t_i | \theta, s_i, \sigma_i) &= \int p(M_i, t_i, d_i | \theta, s_i, \sigma_i)\, d(d_i) \\ &= \int p(M_i | t_i, d_i, \theta, s_i, \sigma_i)\, p(t_i, d_i | \theta, s_i, \sigma_i)\, d(d_i) \\ &= \int p(M_i | d_i, \theta, s_i)\, p(t_i | d_i, \sigma_i)\, \pi(d_i)\, d(d_i) \end{aligned} \tag{3}$$

Le passage de la deuxième égalité à la troisième s'explique par le fait que la loi conditionnelle de M_i sachant (d_i, θ, s_i) ne dépend pas de (t_i, σ_i) et celle de t_i sachant (d_i, σ_i) ne dépend pas de (θ, s_i). Par la formule de Bayes, nous avons également :

$$\pi(d_i|t_i, \sigma_i) \propto p(t_i|d_i, \sigma_i) = \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{d_i - t_i}{\sigma_i}\right)^2\right] \tag{4}$$

Il en découle que la loi *a posteriori* $\pi(d_i|t_i, \sigma_i)$ est la loi gaussienne $N(t_i, \sigma_i^2)$. L'égalité (3) devient alors

$$\begin{aligned} p(M_i, t_i|\theta, s_i, \sigma_i) &= \int p(M_i|d_i, \theta, s_i)\, \pi(d_i|t_i, \sigma_i)\, d(d_i) \\ &= \mathbb{E}_{\pi(d_i|t_i,\sigma_i)}\left[p(M_i|d_i, \theta, s_i)\right] \end{aligned} \tag{5}$$

Un estimateur de Monte Carlo de cette densité jointe est donnée alors par

$$\begin{aligned} \widehat{p}_K(M_i, t_i|\theta, s_i, \sigma_i) &= \frac{1}{K}\sum_{k=1}^{K} p(M_i|\theta, d_i^k, s_i) \\ &= \frac{1}{s_i\sqrt{2\pi}}\frac{1}{K}\sum_{k=1}^{K} exp\left\{-\frac{1}{2}\frac{\left(M_i - g(d_i^k, \theta)\right)^2}{s_i^2}\right\} \end{aligned} \tag{6}$$

avec $\left(d_i^k\right)$, $k = \{1, \cdots, K\}$ i.i.d de loi $N(t_i, \sigma_i^2)$.

Finalement, en utilisant (5) et (6) et en notant $\underline{M} = (M_1, \cdots, M_n)$, $\underline{s} = (s_1, \cdots, s_n)$, $\underline{t} = (t_1, \cdots, t_n)$, $\underline{\sigma} = (\sigma_1, \cdots, \sigma_n)$ et $\underline{d} = (d_1, ..., d_n)$ les vecteurs contenant les différentes données et variables, la vraisemblance du modèle est alors donnée par

$$\begin{aligned} \mathcal{L}(\theta) &= p(\underline{M}, \underline{t}|\theta, \underline{s}, \underline{\sigma}) \\ &= \left(\frac{1}{\sqrt{2\pi}}\right)^n \prod_{i=1}^{n}\frac{1}{s_i}\mathbb{E}_{\pi(d_i|t_i,\sigma_i)}\left[\exp\left\{-\frac{1}{2}\frac{\left(M_i - g(d_i, \theta)\right)^2}{s_i^2}\right\}\right] \end{aligned} \tag{7}$$

et son estimateur par, pour tout entier $K \geq 1$,

$$\widehat{\mathcal{L}}_K(\theta) = \left(\frac{1}{\sqrt{2\pi}}\right)^n \frac{1}{K^n}\prod_{i=1}^{n}\frac{1}{s_i}\sum_{k=1}^{K} exp\left\{-\frac{1}{2}\frac{\left(M_i - g(d_i^k, \theta)\right)^2}{s_i^2}\right\} \tag{8}$$

avec $\left(d_i^k\right)$, $k = \{1, \cdots, K\}$ et $i = \{1, \cdots, n\}$ indépendantes et pour tout $i \in \{1, \cdots, n\}$, $\left(d_i^k\right)_{1\leq k\leq K}$ i.i.d de loi $N(t_i, \sigma_i^2)$.

Dans le cadre bayésien, l'inférence et la prédiction du modèle décrit par le système d'équations (1) sont basées sur la loi *a posteriori* $\pi(\theta|\mathcal{D})$ sur les paramètres du réseau. Cette loi s'obtient par la formule de Bayes :

$$\pi(\theta|\mathcal{D}) = \frac{\mathcal{L}(\theta)\cdot\pi(\theta)}{m(\underline{M}, \underline{t}|\underline{s}, \underline{\sigma})} \tag{9}$$

avec $m(\underline{M}, \underline{t}|\underline{s}, \underline{\sigma}) = \int \mathcal{L}(\theta) \cdot \pi(\theta) d\theta$. Cette constante de normalisation étant en général difficile à calculer, les méthodes de Monte Carlo par Chaînes de Markov (MCMC) sont alors utilisées dans les cas de modèles bayésiens classiques afin de générer un échantillon suivant la loi *a posteriori*. Cependant, les réseaux de neurones ont souvent un nombre de paramètres très élevé, ce qui rend l'utilisation des algorithmes MCMC impossible dans un laps de temps raisonnable. La solution est alors de recourir à l'inférence variationnelle :

- son principe : remplacer la loi *a posteriori* $\pi(\theta|\mathcal{D})$ par une loi $\pi(\theta|\psi)$ « plus simple à manipuler » appelée distribution variationnelle,
- le but sera alors de trouver le paramètre ψ tel que la dissimilarité entre ces deux lois soit minimale,
- pour ce faire, on minimise la divergence de Kullback Leibler, notée $KL(\cdot \,||\, \cdot)$, entre les deux distributions :

$$\begin{aligned} KL\big(\pi(\theta|\psi) \,||\, \pi(\theta|\mathcal{D})\big) &= \int \pi(\theta|\psi) \, \log \frac{\pi(\theta|\psi)}{\pi(\theta|\mathcal{D})} \, d\theta \\ &= \mathbb{E}_{\pi(\theta|\psi)} \left[\log \frac{\pi(\theta|\psi)}{\pi(\theta|\mathcal{D})} \right] \\ &= KL\big(\pi(\theta|\psi) \,||\, \pi(\theta)\big) - \mathbb{E}_{\pi(\theta|\psi)} \left[\log \left(\mathcal{L}(\theta) \right) \right] + \log \left(m(\underline{M}, \underline{t}|\underline{s}, \underline{\sigma}) \right) \end{aligned} \tag{10}$$

L'équation (10) montre alors que minimiser la divergence de Kullback Leibler entre la loi *a posteriori* et la loi variationnelle revient tout simplement à minimiser la fonction de perte $\ell(\psi)$ donnée par

$$\ell(\psi) = KL\big(\pi(\theta|\psi) \,||\, \pi(\theta)\big) - \mathbb{E}_{\pi(\theta|\psi)} \left[\log \left(\mathcal{L}(\theta) \right) \right] \tag{11}$$

L'opposé de cette fonction de perte est connu dans la littérature sous l'acronyme **ELBO** pour *Evidence Lower Bound*, voir par exemple dans Jaakkola and Jordan (2000). Le livre Bishop (2006) contient un bon chapitre introductif sur l'approche variationnelle et Blei et al. (2017) en donne une revue détaillée.

Nous pouvons alors recourir à une double approximation par Monte Carlo pour calculer un estimateur $\widetilde{\ell}_{K,Q}(\psi)$ de la fonction de perte donnée par (11) : on remplace d'abord la vraisemblance $\mathcal{L}(\theta)$ par son estimateur $\widehat{\mathcal{L}}_K(\theta)$ donné par (8), ensuite on estime $\mathbb{E}_{\pi(\theta|\psi)} \left[\log \left(\widehat{\mathcal{L}}_K(\theta) \right) \right]$ en simulant un Q-échantillon (θ_q), $q = \{1, \cdots, Q\}$ suivant la loi variationnelle $\pi(\theta|\psi)$. On obtient alors

$$\widetilde{\ell}_{K,Q}(\psi) = C_n + KL\big(\pi(\theta|\psi) \,||\, \pi(\theta)\big) - \sum_{i=1}^{n} \frac{1}{Q} \sum_{q=1}^{Q} \log \left(\frac{1}{K} \sum_{k=1}^{K} exp \left\{ -\frac{1}{2} \frac{\left(M_i - g(d_i^k, \theta_q)\right)^2}{s_i^2} \right\} \right) \tag{12}$$

avec $C_n = n \log(\sqrt{2\pi}) + \sum_{i=1}^{n} \log(s_i)$ une constante.

Et en appliquant l'inégalité de Jensen à la fonction $-\log$ qui est convexe, on obtient la majoration

$$\widetilde{\ell}_{K,Q}(\psi) \leq \widehat{\ell}_{K,Q}(\psi) \tag{13}$$

dans laquelle l'expression analytique de la fonction majorante $\widehat{\ell}_{K,Q}(\psi)$ est donnée par

$$\widehat{\ell}_{K,Q}(\psi) = C_n + KL\big(\,\pi(\theta|\psi)\,||\,\pi(\theta)\,\big) + \frac{1}{2}\sum_{i=1}^{n}\frac{1}{Q}\sum_{q=1}^{Q}\frac{1}{K}\sum_{k=1}^{K}\frac{\big(M_i - g(d_i^k,\theta_q)\big)^2}{s_i^2} \tag{14}$$

C'est cette dernière fonction $\widehat{\ell}_{K,Q}$ que nous allons utiliser comme fonction de perte. En effet, l'estimateur $\widetilde{\ell}_{K,Q}$ donné par (12) est certes calculable mais nécessiterait une bouche d'apprentissage conçue sur mesure car ce n'est pas une fonction standard qu'on retrouve dans les librairies classiques de deep learning. Son évaluation peut aussi avoir un coût de calcul non négligeable selon l'efficacité de son implémentation. A l'opposé, avec le choix de $Q = 1$, utiliser $\widehat{\ell}_{K,Q}$ comme fonction de perte lors de l'entraînement du réseau est équivalent à minimiser l'erreur quadratique moyenne (MSE) pondérée d'un réseau avec variable explicative non entachée d'erreur et entrainé sur $n \cdot K$ données $\mathcal{D}_{nK} = \big\{(d_i^k, M_i, s_i), i = 1, \cdots, n \texttt{ et } k = 1, \cdots, K\big\}$ où les d_i^k ont été générées en amont suivant les lois $N(t_i, \sigma_i^2)$. Et à cette MSE pondérée est ajouté un terme de régularisation donné par la divergence de Kullback Leibler entre la loi variationnelle et la loi *a priori*. De plus, l'entraînement peut être séquentiel ou distribué ; ce qui donne deux stratégies d'apprentissage présentées dans la section 3 ci-dessous. Cependant, minimiser la fonction de remplacement $\widehat{\ell}_{K,Q}$ ne garantit pas forcément l'obtention d'une solution optimale pour la vraie fonction de perte $\widetilde{\ell}_{K,Q}$ qu'elle majore. Nous avons donc testé l'utilisation de cette approche sur des données simulées pour évaluer sa capacité à estimer la vraie fonction de régression. Les résultats obtenus sont l'objet de la section 4.

La divergence entre la loi variationnelle et la loi *a priori* $KL\big(\,\pi(\theta|\psi)\,||\,\pi(\theta)\,\big)$ qui apparaît comme terme de régularisation de la fonction de perte dans (11), (12), et (14) peut être approchée numériquement par Monte Carlo comme l'espérance, sous la loi variationnelle, de $\log\dfrac{\pi(\theta|\psi)}{\pi(\theta)}$. Dans ce travail, nous avons choisi des lois variationnelles gaussiennes indépendantes sur les poids et biais du réseau de neurones. Dans ce cas gaussien pour la loi variationnelle et la loi *a priori* sur les paramètres, on peut même donner une expression analytique explicite de la divergence de Kullback Leibler entre les deux lois. Si on réécrit $\theta = (\theta_1, \cdots, \theta_S)$ et $\psi = (\psi_1, \cdots, \psi_S) = ((\psi_{1,1}, \psi_{1,2}), \cdots, (\psi_{S,1}, \psi_{S,2}))$, où S représente le nombre de paramètres du réseau, on peut montrer que

$$KL\big(\,\pi(\theta|\psi)\,||\,\pi(\theta)\,\big) = -\frac{S}{2} + \sum_{s=1}^{S}\left[\frac{\psi_{s,1}^2}{2} + \frac{\psi_{s,2}^2}{2} - \log\left(\psi_{s,2}\right)\right] \tag{15}$$

Maintenant que tous les termes de la fonction de perte sont facilement calculables, l'entraînement du réseau peut se faire comme d'habitude par rétro-propagation des gradients en utilisant par exemple le très populaire astuce de ré-paramétrisation utile dans le contexte de l'approche variationnelle et conduisant à l'algorithme Bayes-by-Backprop introduite par Blundell et al. (2015). Quant au choix de $Q = 1$ mentionné ci-haut, Kingma and Welling (2014) expliquent qu'il est valable à condition que la taille des mini batchs utilisés lors de l'entraînement du réseau soit grande, par exemple 100.

3 Stratégies d'apprentissage

Nous venons de voir que l'utilisation de la fonction de perte définie par (14) permet d'entrainer le réseau sur les données $\mathcal{D}_{nK} = \left\{(d_i^k, M_i, s_i), i = 1, \cdots, n \texttt{ et } k = 1, \cdots, K\right\}$ où les d_i^k ont été générées en amont suivant les lois $N(t_i, \sigma_i^2)$. Nous avons alors deux possibilités pour l'entraînement du réseau : un apprentissage séquentiel ou un apprentissage distribué.

Apprentissage séquentiel

En utilisant les différentes lois *a priori* et *a posteriori* spécifiées précédemment ainsi que la distribution variationnelle gaussienne, l'apprentissage séquentiel se fait en deux étapes simples :

— étape 1 : choisir une valeur de K et générer les données $\left\{(d_i^k, M_i, s_i)\right\}$, $i = 1, \cdots, n$ et $k = 1, \cdots, K$, où les $d_i^k \sim N(t_i, \sigma_i^2)$.

— étape 2 : entraîner un réseau bayésien de manière habituelle avec ces données en choisissant comme fonction de perte la MSE pondérée régularisée par la divergence de Kullback Leibler. Ceci se fait très simplement en utilisant les librairies de deep learning classiques, par exemple Keras et Tensorflow-Probability.

Apprentissage distribué

Le fait de passer d'un jeu des données de taille à n à un jeu des données de taille d'ordre $n \cdot K$ peut-être couteux en temps de calcul, surtout si n et K sont grands. L'idée est alors de subdiviser le jeu des données $\left\{(d_i^k, M_i, s_i)\right\}$ en K jeux de données pour entraîner K réseaux de neurones distincts et indépendants mais possédant la même architecture (nombre de couches, neurones par couche et fonctions d'activation). Le réseau 1 utilise les données $\{(d_i^1, M_i, s_i)\}$, le réseau 2 les données $\{(d_i^2, M_i, s_i)\}$, ainsi de suite jusqu'au réseau K qui va utiliser les données $\left\{(d_i^K, M_i, s_i)\right\}$ avec $i = 1, \cdots, n$ pour chaque jeu de données. L'entraînement de chaque réseau se fait alors suivant les étapes de l'apprentissage séquentiel avec une petite modification sur la loi à priori $\pi(\theta)$: au lieu de prendre des gaussiennes centrées réduites, on prend des lois gaussiennes centrées mais de variance K^{-1} pour permettre la reconstruction de la loi *a posteriori* initiale, et donc de la loi variationnelle dans notre cas. Ces réseaux étant indépendants, ils peuvent être entrainés en parallèle selon les ressources de calcul disponibles.

Une fois les réseaux entrainés, il faut recombiner leurs lois variationnelles pour former la loi variationnelle du réseau unique qui aurait pu être entraîné sur toutes ces données en mode apprentissage séquentiel. Alors pour tout paramètre θ_s du réseau unique, sa distribution variationnelle reconstruite à partir de celles de K réseaux est la loi normale de moyenne $\psi_{s,1}$ et de variance $\psi_{s,2}^2$ avec

$$\psi_{s,2}^2 = \left(\sum_{k=1}^{K} \frac{1}{\psi_{s,2,k}^2}\right)^{-1} \quad \texttt{et} \quad \psi_{s,1} = \psi_{s,2}^2 \sum_{k=1}^{K} \frac{\psi_{s,1,k}}{\psi_{s,2,k}^2}$$

où pour tout $1 \leq k \leq K$, $\psi_{s,1,k}$ et $\psi_{s,2,k}^2$ représentent respectivement la moyenne et la variance

de la loi variationnelle gaussienne du paramètre θ_s dans le k $^{\text{ième}}$ réseau de neurone. Cette méthode est par exemple utilisée par Huang and Gelman (2005) pour reconstruire la loi *a posteriori* par approximation gaussienne lors d'un apprentissage distribué.

Pour améliorer les performances du réseau en termes de temps de calcul et réduire aussi le nombre de paramètres qui croit très vite avec la taille du réseau dans le cas bayésien, une méthode pratique mentionnée dans Jospin et al. (2022) consiste à ne faire du bayésien que sur les dernières couches du réseau. Nous avons appliqué cette méthode dans Ashuza Cirumanga et al. (2023) pour proposer une méthode de calibration des âges carbone 14. Si on veut appliquer l'apprentissage distribué dans ce cas de figure, on peut d'abord entraîner un réseau non bayésien une fois sur toutes les données. Ensuite il suffit de figer les paramètres de toutes les couches non bayésiennes et faire de l'apprentissage distribué pour les couches bayésiennes.

4 Expérimentations numériques et conclusion

Nous appliquons l'approche proposée sur des données simulées. Le but est de vérifier si l'on arrive à estimer la vraie fonction de régression au cœur du processus de génération des données. Toutefois, nous veillons à ce que les données simulées aient des caractéristiques similaires (variation de la fonction, structure et taille des erreurs par rapport aux variables explicative et à expliquer) aux données de datation carbone 14 afin de pouvoir appliquer la même approche à l'estimation de courbe de calibration des âges carbone 14 pour les périodes où les dates sont entachées d'erreurs ; ce qui complétera le travail présenté dans Ashuza Cirumanga et al. (2023) qui portait sur l'estimation de cette courbe pour les périodes de temps où la chronologie est connue de manière absolue.

La fonction g utilisée pour générer les données est alors donnée par :

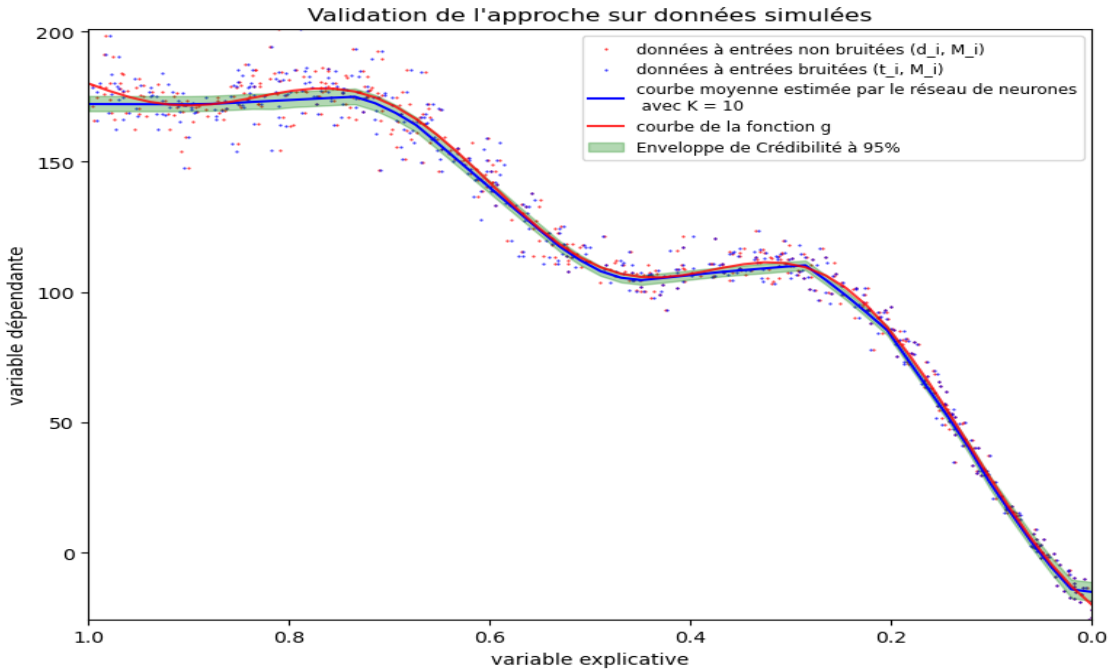
$$\forall x \in [0,1], g(x) = \frac{b-a}{2}\left[x + \frac{1}{3}\left(\sin^2(2\pi x) + \cos\left((1-2x)\frac{\pi}{2}\right) + x^3\cos^2(2\pi x)\right)\right] + a$$

Les données d'apprentissage $\mathcal{D} = \{(t_i, \sigma_i, M_i, s_i), i = 1, \cdots, n\}$ sont alors obtenues comme suit : pour tout $i \in \{1, \cdots, n\}$:

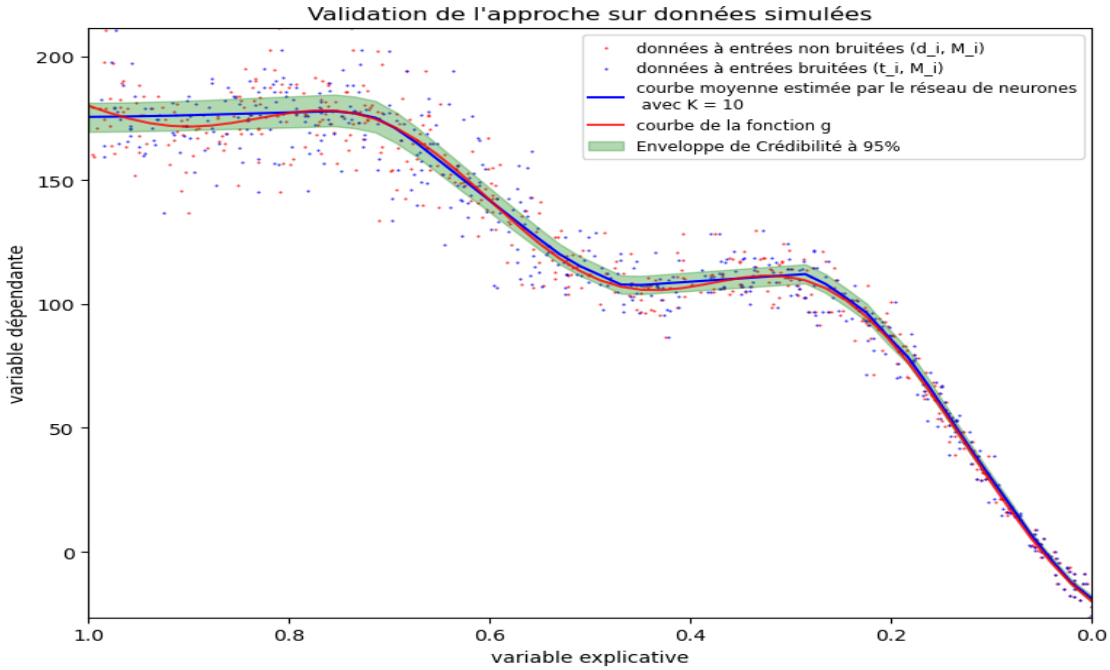
- on tire la vraie valeur d_i de la variable explicative suivant une loi uniforme sur $[0,1]$,
- $\sigma_i = (\alpha + \beta v_i)|d_i|$ avec v_i tiré uniformément sur $[0,1]$,
- $t_i = d_i + \sigma_i u_i$ avec $u_i \sim N(0,1)$,
- $s_i = (\lambda + \kappa w_i)|g(d_i)|$ avec w_i tiré uniformément sur $[0,1]$, et
- $M_i = g(d_i) + s_i\,\varepsilon_i$ avec $\varepsilon_i \sim N(0,1)$.

Pour générer les données, nous avons fixé $a = -20$ et $b = 280$. α, β, λ et κ sont des réels choisis dans $[0,1]$ de telle sorte que les erreurs sur les variables explicative et à expliquer soient de taille voulue (en pourcentage par rapport à la variable concernée).

Sur la figure 1, le graphique (1a) montre le résultat obtenu pour l'estimation de la fonction g quand l'erreur commise sur la variable explicative est de l'ordre de 0.3% à 2.3% et celle sur la variable à expliquer de 2% à 7%. Ces ordres d'erreur correspondent à ceux qui sont observés dans les données de datation des âges carbone 14 au delà des 12 derniers millénaires. Le graphique (1b) montre la même chose pour des ordres d'erreur un peu plus grand : 3% à 5%



(a) $\alpha = 0.003$, $\beta = 0.02$, $\lambda = 0.02$, $\kappa = 0.05$



(b) $\alpha = 0.03$, $\beta = 0.02$, $\lambda = \kappa = 0.05$

FIGURE 1 – Estimation de g avec un réseau de neurones hybride à trois couches cachées contenant respectivement 20, 260 et 240 neurones pour les trois couches successives. Seule la couche de sortie est bayésienne. $n = 1500$ observations utilisées pour l'apprentissage.

pour la variable explicative et 5% à 10% pour la variable à expliquer. Dans les deux cas, la qualité d'estimation de la courbe est satisfaisante et l'approche peut donc être utilisée pour

estimer la courbe de calibration des âges carbone 14. Le graphique (1b) illustre bien aussi l'impact des erreurs en entrée sur la largeur de l'enveloppe de crédibilité obtenue autour de la courbe.

Références

Ashuza Cirumanga, D., Philippe, A., and Guérin, G. (2023). Approche bayésienne et réseaux de neurones appliqués à la calibration des âges carbone 14. In *communication lors des 54es Journées de la Statitistique de la SFdS*.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference : A review for statisticians. *Journal of the American statistical Association*, 112(518) :859–877.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight uncertainty in neural network. In *International conference on machine learning*, pages 1613–1622. PMLR.

Carroll, R. J., Ruppert, D., Stefanski, L. A., and Crainiceanu, C. M. (2006). *Measurement error in nonlinear models : a modern perspective*, volume 2. Chapman and Hall/CRC.

Dillon, J. V., Langmore, I., Tran, D., Brevdo, E., Vasudevan, S., Moore, D., Patton, B., Alemi, A., Hoffman, M., and Saurous, R. A. (2017). Tensorflow distributions. *arXiv preprint arXiv :1711.10604*.

Heaton, T. J., Blaauw, M., Blackwell, P. G., Ramsey, C. B., Reimer, P. J., and Scott, E. M. (2020). The intcal20 approach to radiocarbon calibration curve construction : a new methodology using bayesian splines and errors-in-variables. *Radiocarbon*, 62(4) :821–863.

Huang, Z. and Gelman, A. (2005). Sampling for bayesian computation with large datasets. *Available at SSRN 1010107*.

Jaakkola, T. S. and Jordan, M. I. (2000). Bayesian parameter estimation via variational methods. *Statistics and Computing*, 10 :25–37.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2021). *An introduction to statistical learning : With applications in R*. Springer.

Jospin, L. V., Laga, H., Boussaid, F., Buntine, W., and Bennamoun, M. (2022). Hands-on bayesian neural networks—a tutorial for deep learning users. *IEEE Computational Intelligence Magazine*, 17(2) :29–48.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*.

Martin, J. and Elster, C. (2023). Aleatoric uncertainty for errors-in-variables models in deep regression. *Neural Processing Letters*, 55(4) :4799–4818.