Q-Net: A Network for Low-dimensional Integrals of Neural Proxies

Kartic Subr – CGF 2021

Groupe De Lecture - 14/09/2022

Using Neural Networks for Fast Numerical Integration and Optimization

Steffan Lloyd, Rishad Irani, and Mojtaba Ahmadi

IEE Access 8 - 2020

Qu’est-ce qu’un réseau de neurones ?

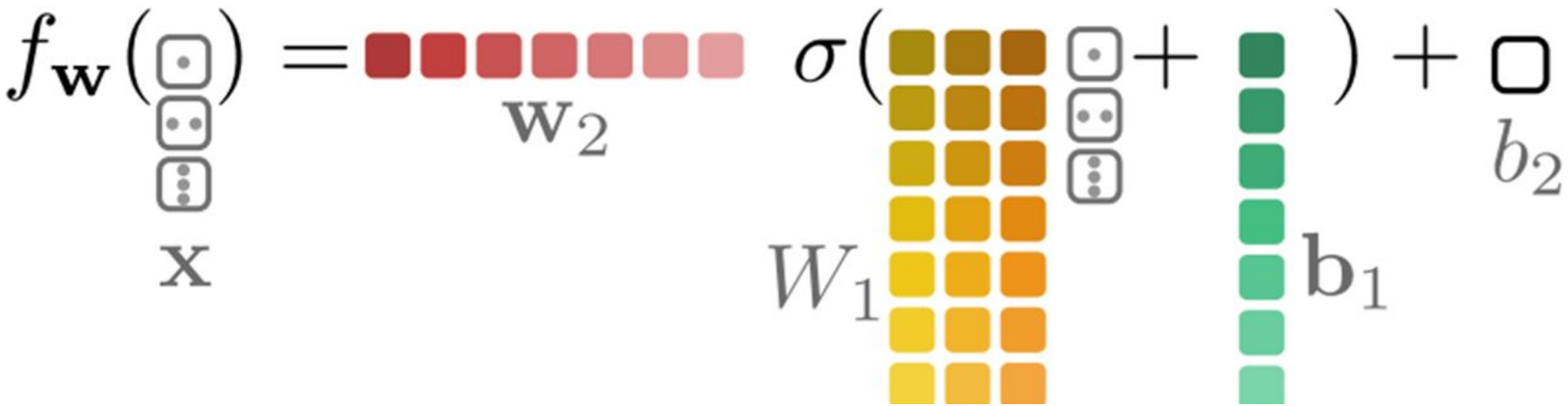


Illustration d’un réseau de neurones feed-forward (FFN).

Théorème d’approximation universelle

Pour toute fonction continue f, il existe un FFN avec une seule couche cachée capable d’être aussi proche que l’on désire de f pour peu que la fonction d’activation soit non polynomiale.

[Cybenko 1989, Hornik 1991, Zhou Lu et al. 2017]

Théorème d'approximation universelle

Pour toute fonction continue f, il existe un FFN avec une seule couche cachée capable d'être aussi proche que l'on désire de f pour peu que la fonction d'activation soit non polynomiale.

Remarques :

- On ne sait pas construire le FFN en question
- On ne sait pas si le réseau de neurones est "tractable" (poids / nombre de neurones tendant vers l'infinie)
- Si on autorise plusieurs couches, on peut borner le nombre de neurones nécessaire pour une RELU

[Cybenko 1989, Hornik 1991, Zhou Lu et al. 2017]

Théorème d'approximation universelle

Pour toute fonction continue f, il existe un FFN avec une seule couche cachée capable d'être aussi proche que l'on désire de f pour peu que la fonction d'activation soit non polynomiale.

Remarques :

- On ne sait pas construire le FFN en question
- On ne sait pas si le réseau de neurones est "tractable" (poids / nombre de neurones tendant vers l'infinie)
- Si on autorise plusieurs couches, on peut borner le nombre de neurones nécessaire pour une RELU

Si l'on connaît f_w telle que $|f_w - f| \leq \epsilon$ alors :

$$\left| \int_{[0,1]^d} f_w - \int_{[0,1]^d} f \right| \leq \int_{[0,1]^d} |f_w - f| \leq \epsilon$$

[Cybenko 1989, Hornik 1991, Zhou Lu et al. 2017]

Théorème fondamental de l'analyse

En 1D, on a le théorème suivant :

$$\int_{[0,1]^1} f(t)\mathrm{d}t = F(1) - F(0)$$

En 2D, le théorème devient :

$$\int_{[0,1]^2} f(t)\mathrm{d}t = F(1,1) - F(0,1) - F(1,0) + F(0,0)$$

En 3D :

$$\int_{[0,1]^3} f(t)\mathrm{d}t = F(1,1,1) - F(0,1,1) - F(1,0,1) + F(0,0,1) - F(1,1,0) + F(0,1,0) + F(1,0,0) - F(0,0,0)$$

1 lorsque le nombre de 1 a la même parité que d, -1 sinon

Fonction polylogarithmique

La fonction polylogarithmique est définie par la série : $Li_s(z) = \sum_{k=0} \frac{z^k}{k^s}$

$$z\frac{\partial \operatorname{Li}_s(z)}{\partial z} = \operatorname{Li}_{s-1}(z)$$

$$\frac{\partial \operatorname{Li}_s(e^\mu)}{\partial \mu} = \operatorname{Li}_{s-1}(e^\mu)$$

$$\int -Li_s\left(-e^{\kappa(z)}\right) \mathrm{d}z = -\left(\frac{\partial \kappa(z)}{\partial z}\right)^{-1} Li_{s+1}\left(-e^{\kappa(z)}\right)$$

$$\operatorname{Li}_1(z) = -\ln(1-z)$$

$$\operatorname{Li}_0(z) = \frac{z}{1-z}$$

$$\operatorname{Li}_{-1}(z) = \frac{z}{(1-z)^2}$$

$$\operatorname{Li}_{-2}(z) = \frac{z(1+z)}{(1-z)^3}$$

$$\operatorname{Li}_{-3}(z) = \frac{z(1+4z+z^2)}{(1-z)^4}$$

$$\operatorname{Li}_{-4}(z) = \frac{z(1+z)(1+10z+z^2)}{(1-z)^5}$$

Q-Net: la construction

Soit σ la fonction sigmoïde, définie par : $\sigma(x) = \frac{1}{1+e^{-x}} = -Li_0(-e^x)$

Un FFN s'écrit : $f_w(x) = W_2\sigma(W_1x + b_1) + b_2 = -W_2Li_0\left(-e^{W_1x+b_1}\right) + b_2$

En intégrant successivement sur les dimensions, on obtient la formule :

$$\int_{[0,1]^d} f_w = W_2v + 2^d b_2$$

$$v^i = 2^d + \frac{1}{\tilde{w}_1^i}\sum_{m=1}^{2^d} \alpha_m \mathrm{Li}_d\left(-\exp\left(S^{m,\cdot}W_1^{i,\cdot T} - b_1^i\right)\right)$$

$$\tilde{w}_1^i = \prod_j W_1^{i,j}, \text{ S are the hypercube vertices, and } \alpha_m \text{ the associated coefficients}$$

Q-Net: la construction

L'expression précédente peut être réécrite comme un réseau de neurones à poids fixes, dont l'input est les poids du réseau de neurone original.

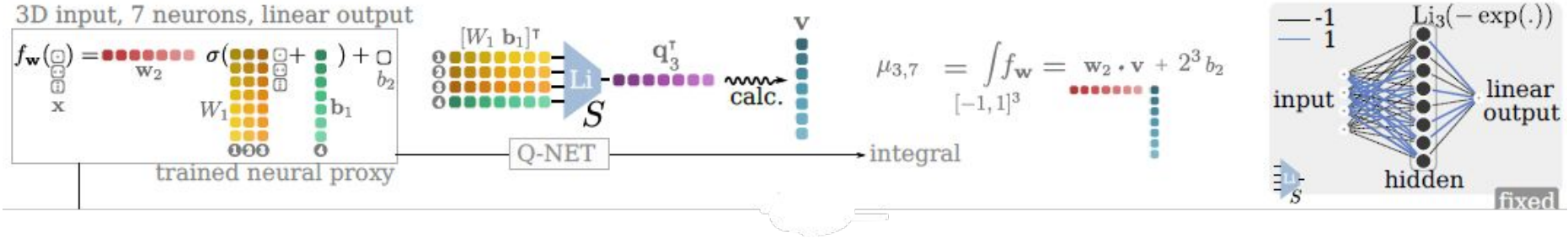
$$v^i = 2^d + \frac{1}{\tilde{w}_1^i} \sum_{m=1}^{2^d} \alpha_m \mathrm{Li}_d \left(-\exp\left(S^{m,\cdot} W_1^{i,\cdot T} - b_1^i \right) \right)$$

On pose : $w_3^i = \alpha^i \ , \ \sigma_d(x) = \mathrm{Li}_d\left(-\exp(x)\right) \ , \ W_4 = S - 1_{2^d}$

Puis : $y^i = [W_1^{i,\cdot} b_1]^T$ $q(y^i) = W_3 \sigma_d(W_4 y^i + 0) + 0$

Pour obtenir : $v^i = 2^d + q(y^i)/\tilde{w}_1^i$

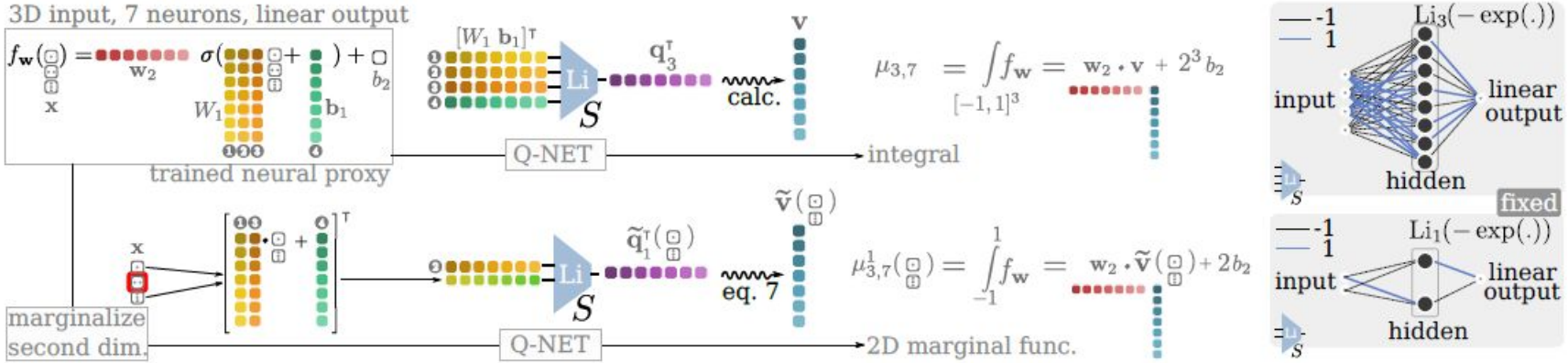
Q-Net: Overview



Opérations avec Q-Net

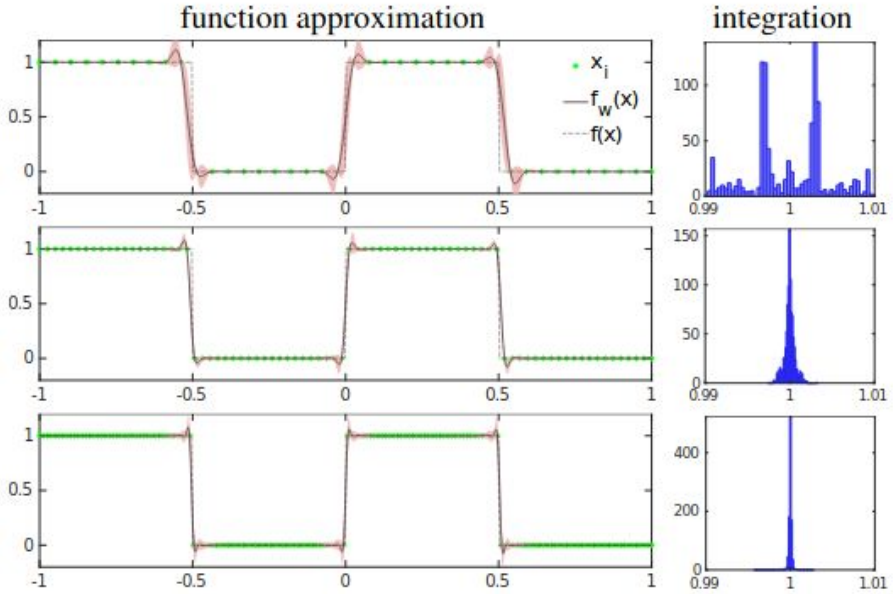
En plus de la simple intégration, plusieurs opérations sont facilement réalisable :

- Transformation affine Mx + c: transformer W_1 par MW_1 et b_1 par $W_1c + b_1$
- Somme : pour deux proxy entraînés, il suffit de concaténer les matrices et bias
- Slicing et Projections : Sélections sur W_1 + mis à jour de b_1
- Intégration sur des sous domaines : remplacer S par les sommets du sous domaine et modifier les constantes



Notes sur l’entraînement

- Le nombre de neurones est un facteur critique. Plusieurs règles empiriques existent… Dans l'article: entre 50 et 200 neurones.
- Le réseau de neurones étant simple, on peut utiliser des méthodes d’optimisation d’ordre 1.5 (Levenberg-Marquardt) voir 2.
- La loss “n’a pas d’importance” sur l'entraînement
- L’impact sur le sampling de la fonction f n’est pas étudié…



Résultats - Fitting test

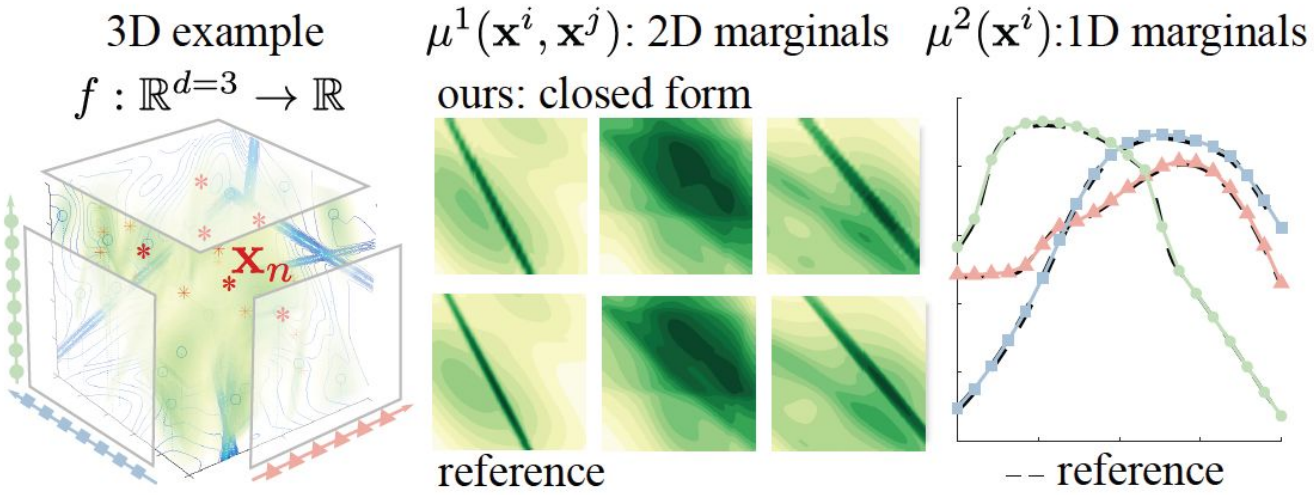


Fig. 5. Results of marginalizing a 3D Gaussian mixture (left) along one dimension (middle) and two dimensions (right). The resulting 2D and 1D marginals are evaluated on grids and compared with reference values.

Résultats - Effet de la dimension

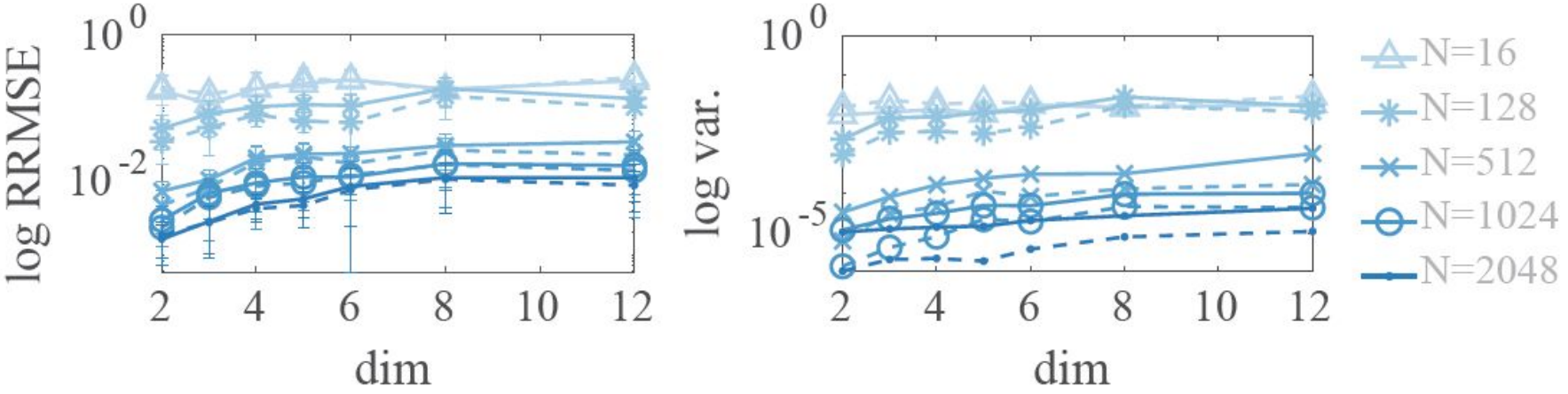


Fig. 7. Increasing dimensionality on HR integrands. Plots of relative error (left) and relative variance (right) averaged over 40 iterations each of 50 random hyperrectangle integrands over dimensions $d = 2, 3, 4, 5, 6, 8, 12$. Although error increases with dimension, the effect is not sustained.

Résultats - Variable de contrôle

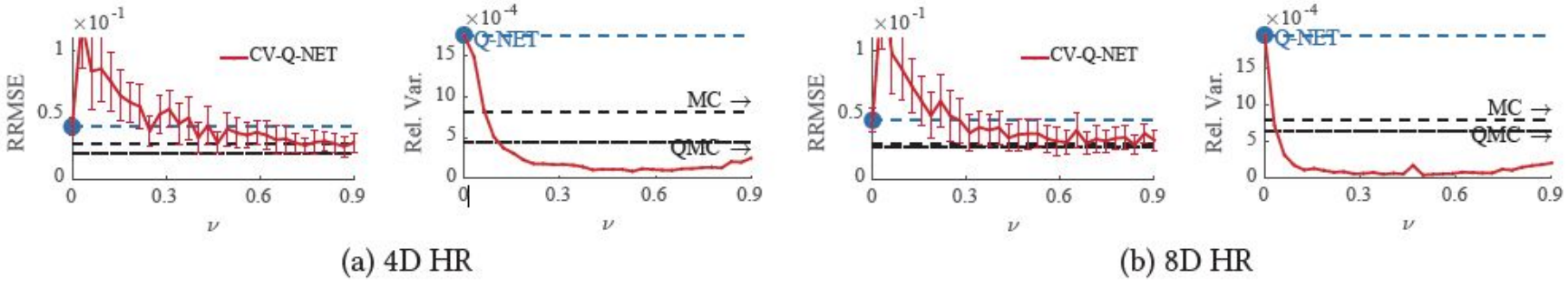
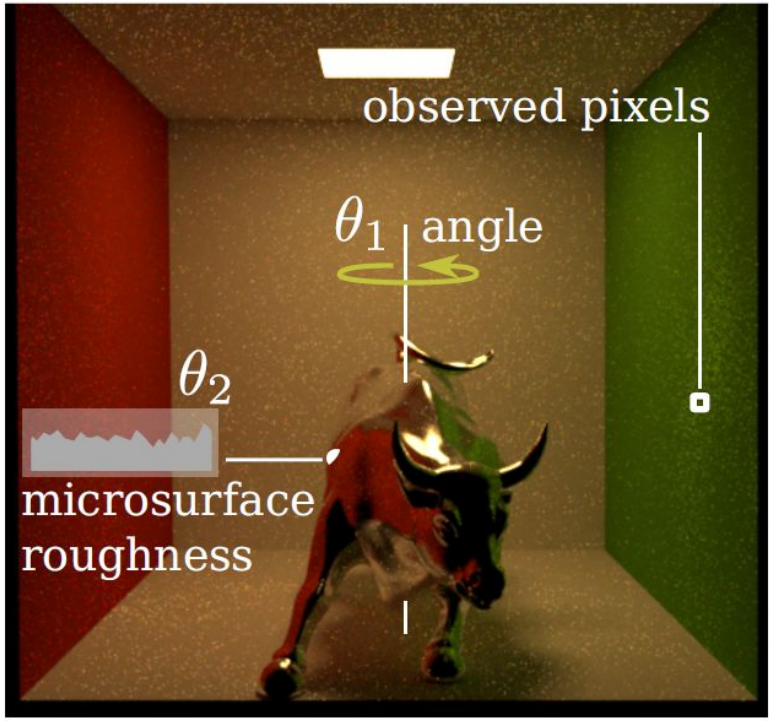
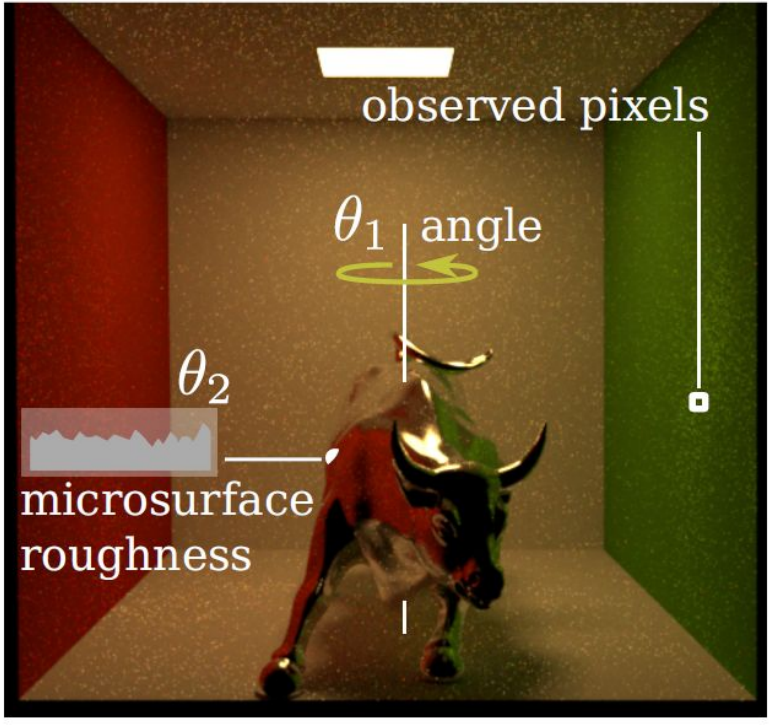


Fig. 9. The proxy can be used as a control variate (CV) by using a fraction (ν) of the samples to integrate the difference $f - f_w$ using MC or QMC. The plots show relative error and variance of the CV estimator (red) compared to Q-NET ($\nu = 0$) and MC and QMC ($\nu = 1$) estimators. They reveal that CV-Q-NET is effective at reducing variance (lower than MC and QMC) for a wide range of values of ν in 4D (a) as well as in 8D (b).

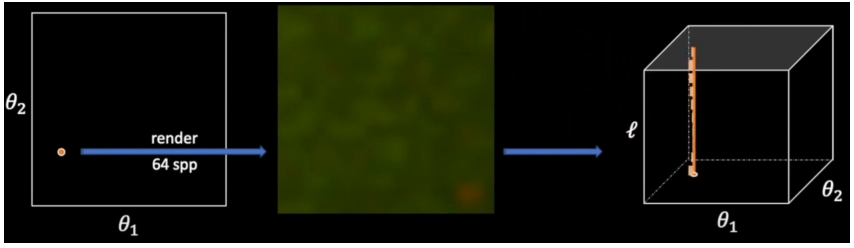
Résultats - Inverse rendering



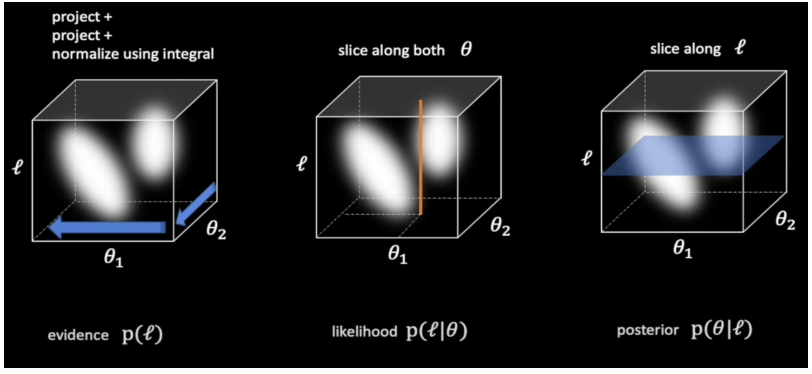
Résultats - Inverse rendering



On observe des patchs de luminance pour plusieurs jeux de paramètres :



On entraîne le proxy à modéliser $P\left[\theta_1, \theta_2, l\right]$



Résultats - Profondeur optique

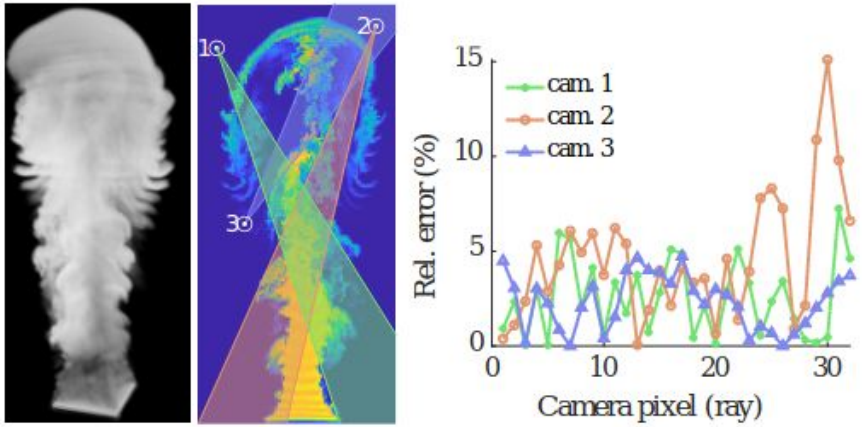
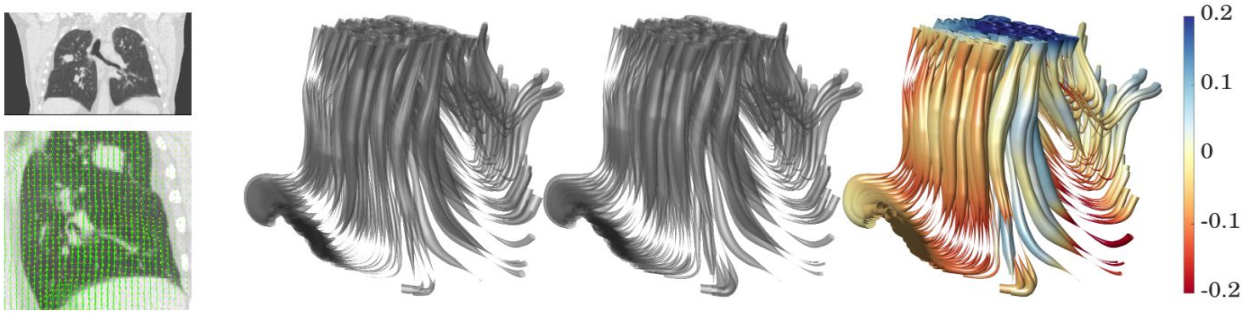


Figure 11: *We fit a proxy to density data from a smoke simulation in Blender and calculate optical depth along rays using a Q-NET. A frame from the simulation is shown rendered using Blender Cycles (left). For the central slice, we use a value proportional to the density as the attenuation coefficient and integrate it along 32 rays each for three virtual camera poses (middle). The plot (right) shows percentage relative error for the rays.*

Résultats - Visualisation du flux

Flux = intégrale de la divergence d'un champ vectoriel



(a) data (b) dil. (ref.) (c) dil. (ours) (d) dil. + flux (ours)

Fig. 13. (a) Images showing the deformation field in a human lung during respiration [Vandemeulebroucke et al. 2007]. (b) Streamtubes for deformation between two frames of the dataset. The thickness of tubes corresponds to local dilatation. (c) Approximation dilatation obtained using a neural proxy. (d) Visualizing local flux (color) calculated by integrating the proxy.

Conclusion

Résumé

- L'architecture QNet permet de faire l’intégration de FFN
- Plusieurs applications en rendering peuvent tirer partie des capacités de QNet : intégrale, projection, gradients, …

Limitations:

- Limité par la dimension du problème, complexité en $O(kd2^d)$
- Les performances sont conditionnées par l’approximation du proxy dont la forme reste simple