

積分型の損失関数を利用する統計的パラメータ推定

奥野彰文^{1,2,3}

¹統計数理研究所, ²総合研究大学院大学, ³理化学研究所 (AIP/CBS)



<https://okuno.net/slides/2025-10-RIMS.pdf>

自己紹介+講演概要

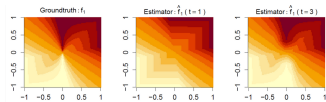
オクノ アキフミ
奥野彰文 博士 (情報学, 京都大学)¹



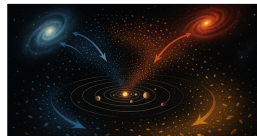
- ▶ 専門：統計的機械学習，特に手法開発とその理論解析.
- ▶ 普段は統計数理研究所 (東京都立川市) にいます.

¹<https://okuno.net>

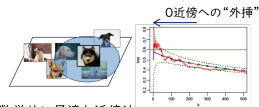
統計～機械学習の何でも屋です



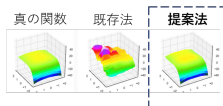
「可逆」という制約を課した推定の最適性
Okuno and Imaizumi (Elec. J. Stat. 2023)



天体の起源を探るクラスタリング:
Hattori, Okuno and Roederer (Astrophysical J. 2023)
Okuno and Hattori (Ann. Ins. Stat. Math. 2025)



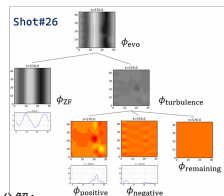
数学的に最適な近傍法
Okuno and Shimodaira (NeurIPS 2020)



類似度ニューラルネットの表現能力とその拡張
Okuno et al. (ICML2018, AISTATS2019)



組成分析における系統的欠測の補完

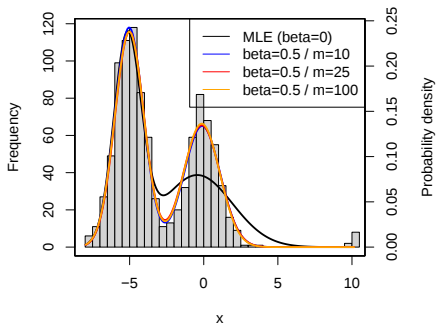


プラズマ乱流の分解:
Okuno et al. (Plas. Fus. Res. 2024)
Okuno and Sasaki (Physics of Plasmas 2025)

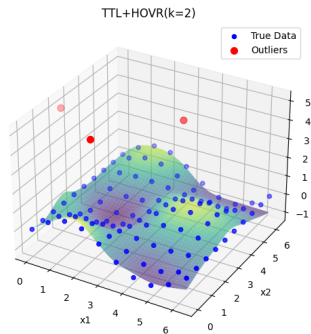
▶ 理論～応用まで，統計関係で何かありましたらお気軽にお声がけください。

今日の講演内容

- ▶ 有限和型 $L_n(\theta) = \frac{1}{n} \sum_{i=1}^n s(z_i; \theta)$ ではなく,
積分型 $L(\theta) = \int s(z; \theta) d\mathbb{P}(z)$ の損失を用いたパラメータ推定.



混合など複雑な分布のロバスト推定
(Okuno, AISM2024)



外れ値に強いニューラルネットの学習
(Okuno and Yagishita, arXiv:2308.02293)

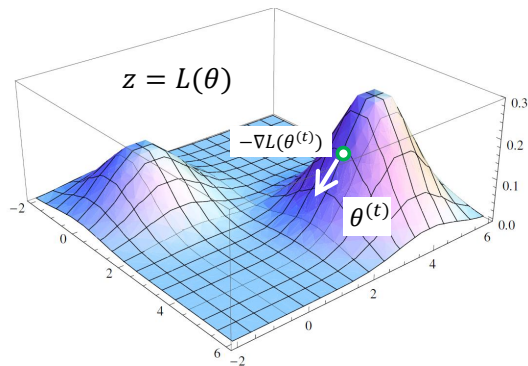
- ① 自己紹介+講演概要
- ② 先に技術的結論から
 - 課題：損失関数が積分型であるつらさ
 - 解決策：確率的最適化
- ③ 応用1：複雑な分布のロバスト推定
 - 外れ値にロバストな分布推定とは
 - 確率的最適化による最適化
- ④ 応用2：外れ値に強いニューラルネットの学習
 - 旧来のロバスト推定とその課題
 - ニューラルネットの変動正則化
 - 確率的最適化を利用したアルゴリズム
- ⑤ より最近の話

先に技術的結論から

損失関数 $L(\theta)$ 最小化により、パラメータ $\hat{\theta}$ を推定する.

勾配法：局所的最速で山を下る方法

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma \nabla_{\theta} L(\theta^{(t)}), \quad t = 1, 2, \dots$$



損失関数が「積分型」だとつらい

積分型の損失関数 $L(\theta) = \int s(z; \theta) d\mathbb{P}(z)$ の勾配法：

$$\theta_*^{(t+1)} \leftarrow \theta_*^{(t)} - \gamma \nabla_{\theta} \int s(z; \theta_*^{(t)}) d\mathbb{P}(z)$$

を数値積分で近似すると：

$$\theta_N^{(t+1)} \leftarrow \theta_N^{(t)} - \gamma \frac{1}{N} \sum_{j=1}^N \nabla_{\theta} s(z_j; \theta_N^{(t)})$$

- ▶ d 次元なら誤差 ε を達成するためにだいたい $N \sim \varepsilon^{-1/d}$.
- ▶ 各ステップで $O(N)$ 回の計算が要る．最適化で T 反復すると全体で $O(\varepsilon^{-1/d} T)^a$.

^a $\varepsilon \approx 0, d \geq 1, T \rightarrow \infty$, なので一般にはやりたくない計算

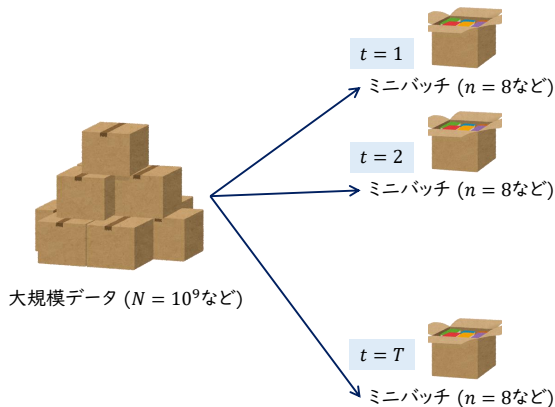
- ▶ なので、スパコンで数値積分をします・・・という話ではありません.



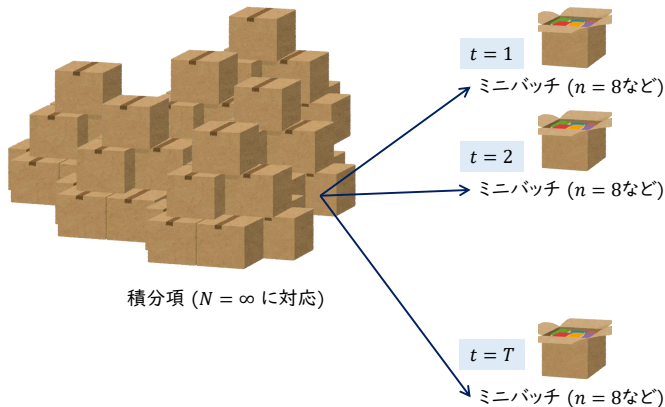
(これはこれでやりたくて、講演の最後に少し話をします)

深層学習などで多用される「確率的最適化」

- ▶ 学習データサイズ N が大きすぎて、各反復にすべてのデータは使わない。



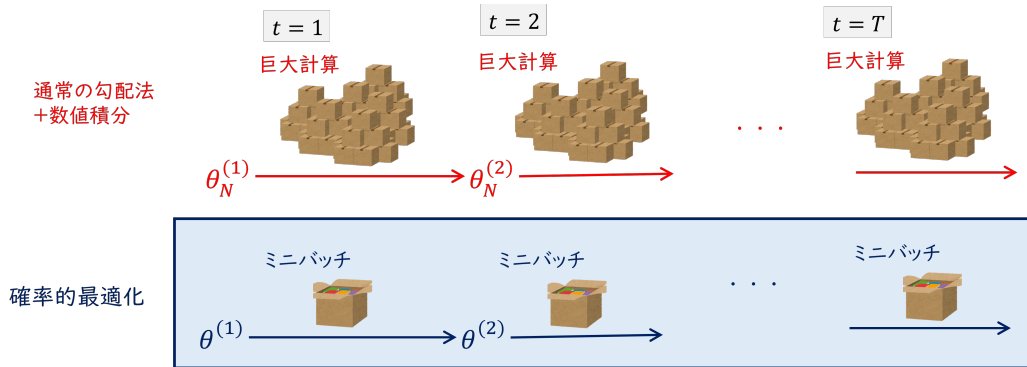
積分は $N = \infty$ と思えばよい.



👍 各ステップで律儀に「無限」を考える必要がない².

²そもそも、どうしても数値計算で無限は扱えない…

つまりどういうことなのか



各反復の計算は軽く、反復全体で実質的に巨大な数値積分を計算する
=毎回大きな計算をする必要がない!!

という話の一般化が75年前

Robbins and Monro (1951)

$\nabla_{\theta} L(\theta)$ の確率的な不偏勾配 $g(\theta)$, つまり

$$\mathbb{E}(g(\theta)) = \nabla_{\theta} L(\theta)$$

を使った確率的勾配法 $\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} g(\theta)$ は $t \rightarrow \infty$ で $L(\theta)$ を最小化する.

- ▶ 通常の勾配法と異なり, $\gamma^{(t)} \searrow 0$ とすることが重要.
- ▶ m 個の乱数 $z_j^{(t)} \sim \mathbb{P}$ を生成して, 以下の更新をするとパラメータが推定できる³:

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \frac{1}{m} \sum_{j=1}^m \nabla_{\theta} s(z_j^{(t)}; \theta^{(t)}).$$

³ m はそこそこ大きくても良いし, 理論的には $m = 1$ でもよい. 全体計算量 $O(\varepsilon^{-1/d} T)$ が $O(T)$ に.

ここまでのまとめ

積分型の損失関数 $L(\theta) = \int s(z; \theta) d\mathbb{P}(z)$ を最小化したければ,

👉 数値積分による近似:

$$\theta_N^{(t+1)} \leftarrow \theta_N^{(t)} - \gamma \frac{1}{N} \sum_{j=1}^N \nabla_{\theta} s(z_j; \theta_N^{(t)}).$$

👉 計算量 $O(\varepsilon^{-1/d} T)$, 👉 誤差 $\rightarrow c_{\varepsilon} > 0$.

👍 確率的な最適化 (Robbins and Monro, 1951):

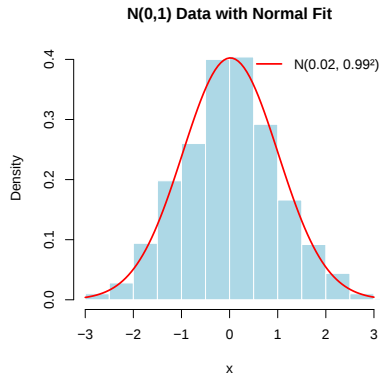
$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \frac{1}{m} \sum_{j=1}^m \nabla_{\theta} s(z_j^{(t)}; \theta^{(t)}), \quad z_j^{(t)} \stackrel{\text{i.i.d.}}{\sim} \mathbb{P}, \quad \gamma^{(t)} \searrow 0.$$

👍 計算量 $O(mT) = O(T)$, 👍 誤差 $\rightarrow^* 0$.

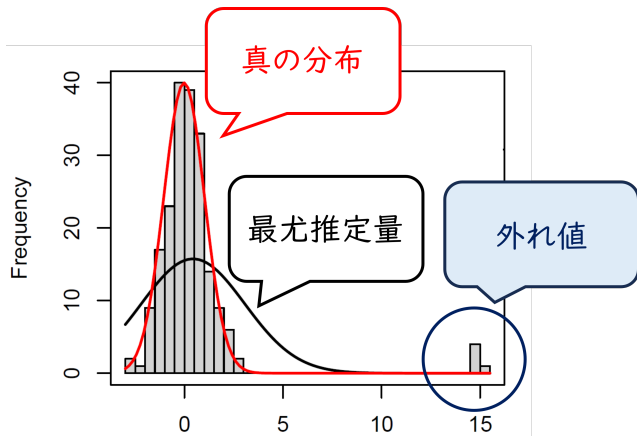
複雑な分布のロバスト推定 (Okuno, AISM2024)

通常の分布推定

- ▶ 観測値 $x_1, x_2, \dots, x_n$ が従う分布 Q を知りたい。
 - ▶ 負の対数尤度 $-\ell(\theta) = -\sum_{i=1}^n \log p_{\theta}(x_i)$ を最小化して $P_{\hat{\theta}} \approx Q$ が求まる。
- 👍 積分が出てこない！



最尤推定は外れ値に弱い



▶ $x_n \rightarrow \infty$ で最尤推定は即座に破綻.

外れ値にロバストな分布推定

冪密度ダイバージェンス (Density power divergence, DPD; Basu et al. 1998):

$$D_{\beta}(\hat{Q}, P_{\theta}) = \underbrace{-\frac{1}{\beta} \frac{1}{n} \sum_{i=1}^n p_{\theta}(x_i)^{\beta} + \int \frac{1}{1+\beta} p_{\theta}(x)^{1+\beta} dx}_{=: L(\theta)} + \text{Const.}$$

を最小化するとよい.

積分が出てきてつらい.

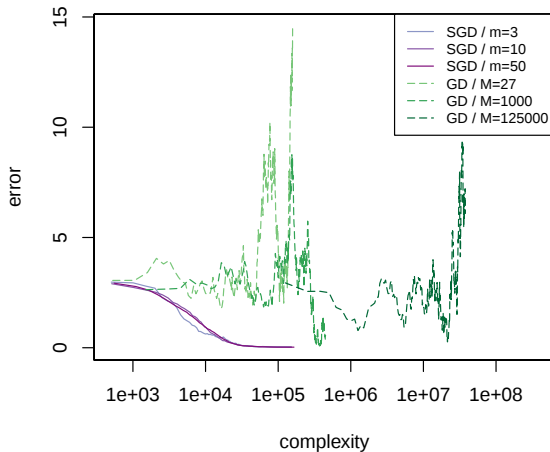
この流れで聞くと自明だが, ロバスト統計界限で膠着状態になっていた.

つらい積分の歴史と今回の研究

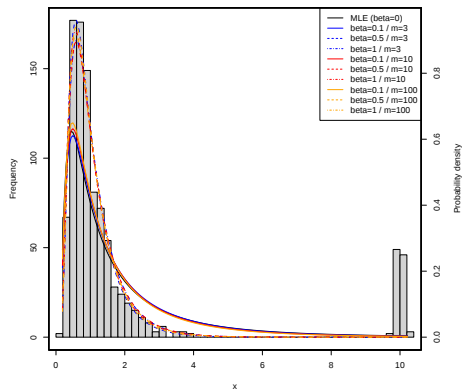
- ▶ 積分が解析的に計算できるのでそれを使う：
 - ▶ 正規分布 (Basu et al., 1998)
 - ▶ 指数分布 (Jones et al., 2001),
 - ▶ 一般化パレート分布 (Juárez and Schucany, 2004),
 - ▶ ワイブル分布 (Basu et al., 2016),
 - ▶ 一般化指数分布 (Hazra, 2022).
- ▶ 解析的に計算できないので数値近似：
 - ▶ 混合正規分布 (Fujisawa and Eguchi, 2006),
 - ▶ ポアソン分布 (Kawashima and Fujisawa, 2019),
 - ▶ 歪正規分布 (Nandy et al., 2022).
- ▶ 各反復でそもそも積分を計算せず，確率的最適化：
 - ▶ 任意の分布 (Okuno, 2024; 今回の研究).

一例として、多変量正規分布 ($d = 3$ 次元) のパラメータ推定

- ▶ 確率的最適化(SGD)の計算量: $t(n + m)$,
- ▶ 通常の勾配法(GD)+数値積分の計算量: $t(n + M)$.



逆ガウス分布, $\xi = 0.1$.



$$p_{\theta}(x) = \sqrt{\frac{\lambda}{2\pi x^3}} \exp\left(-\frac{\lambda(x - \mu)^2}{2\mu^2 x}\right)$$

(*逆ガウス分布の積分項は明示的に計算できない.)

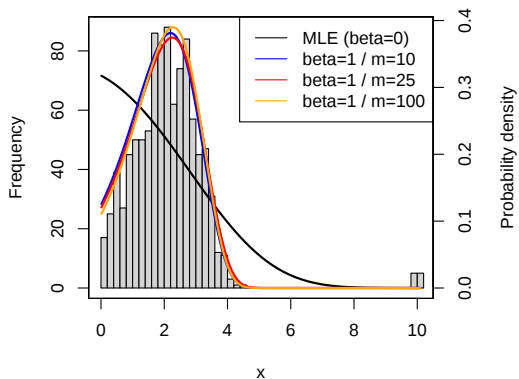


図: ギンペルツ分布

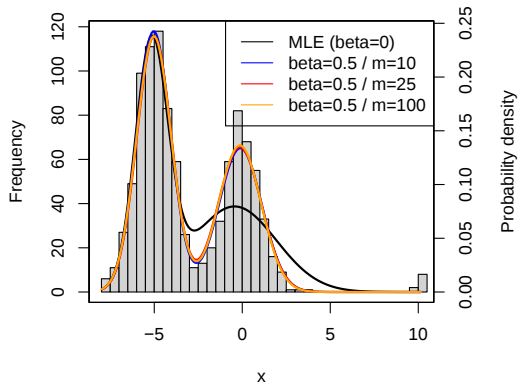


図: 混合正規分布

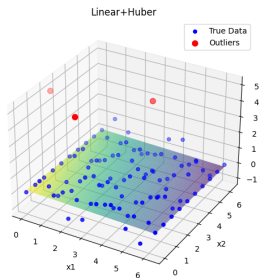
(*ギンペルツ分布も混合正規分布も積分項が明示的に計算できない)

外れ値に強いニューラルネットの学習

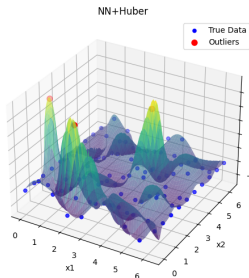
(Okuno and Yagishita, arXiv:2308.02293, in revision)

なにをしたのか

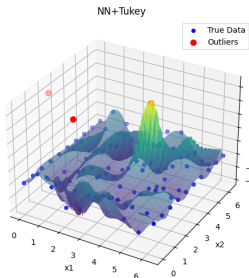
- ▶ 外れ値にロバストかつフレキシブルな予測法の提案.



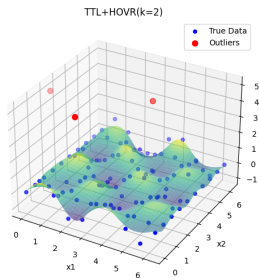
(a) 線形モデル+Huber



(b) NN+Huber



(c) NN+Tukey

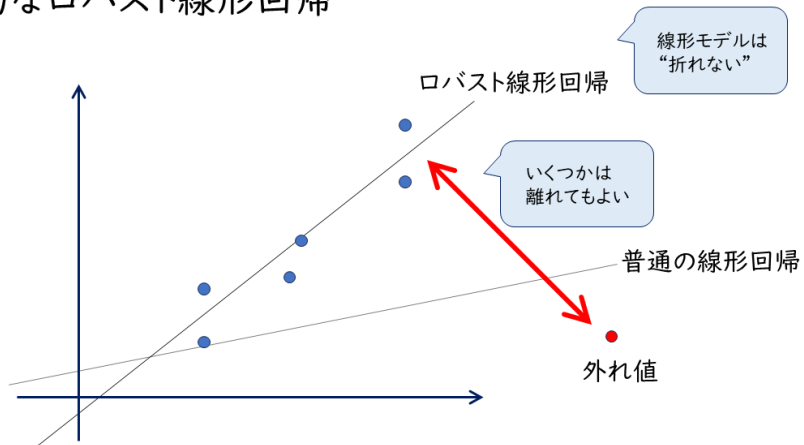


(d) 提案法

図: $f_*(x) = \sin(2x_1) \cos(2x_2)$.

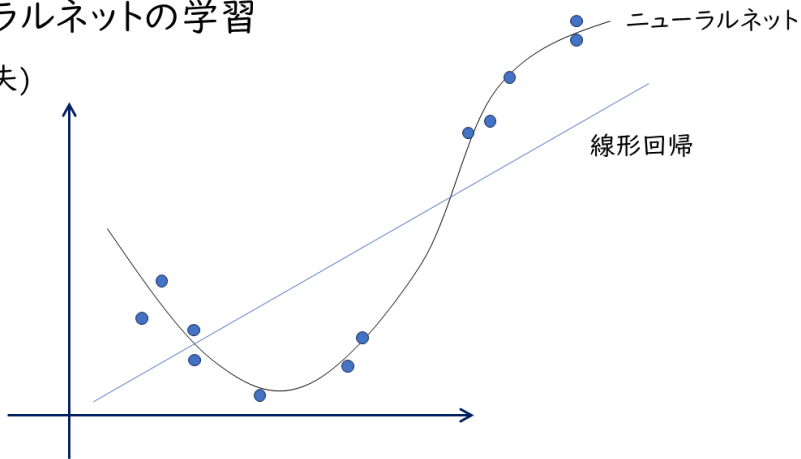
- ▶ 典型的方法「損失が大きなサンプルを捨てる」だけではうまくいかない.

典型的なロバスト線形回帰

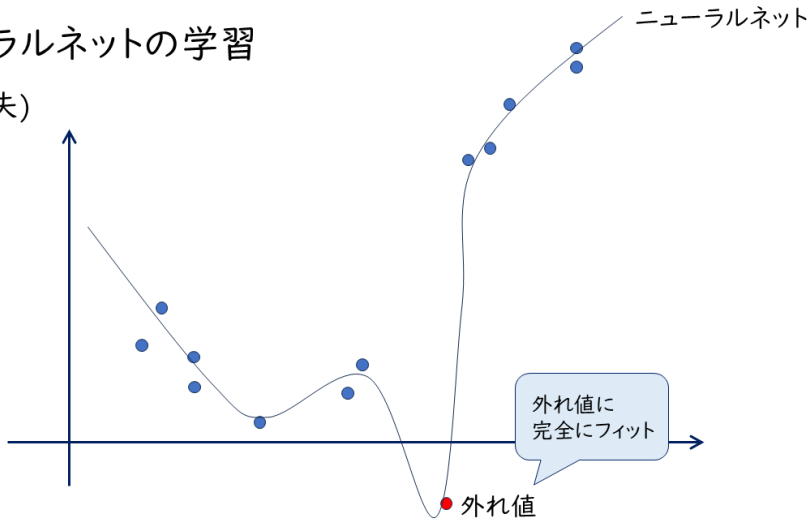


ニューラルネットの学習

(二乗損失)

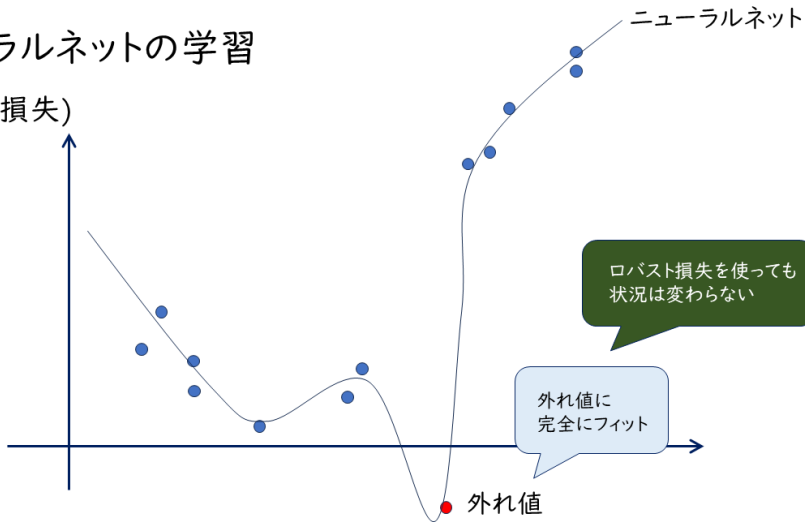


ニューラルネットの学習 (二乗損失)

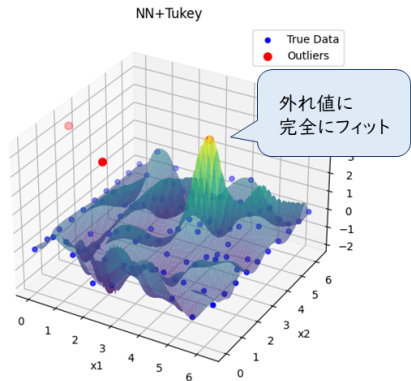
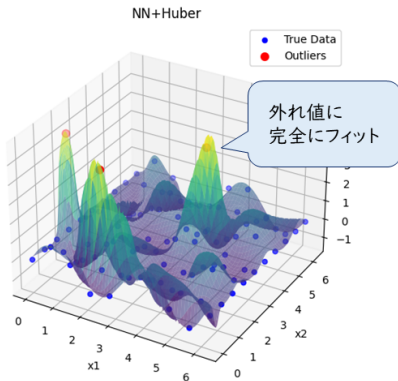


ニューラルネットの学習

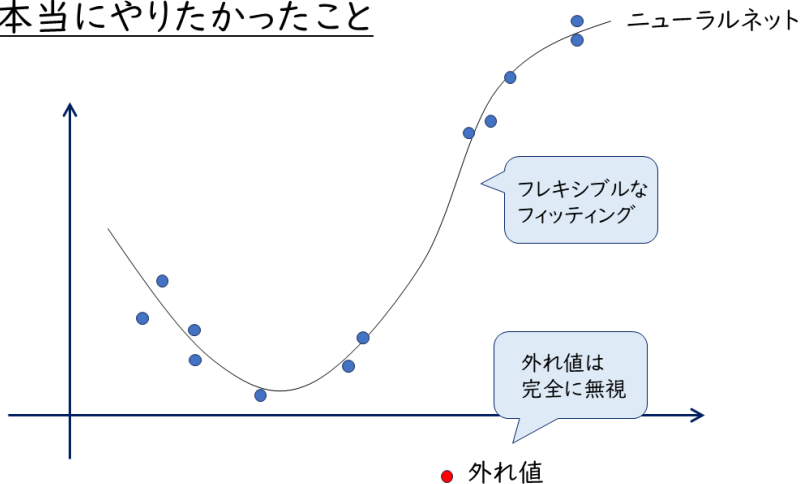
(ロバスト損失)



実際のロバスト損失+NN (+勾配法)



我々が本当にやりたかったこと



つまり何が言いたいのかというと

- ▶ 予測関数 $f_{\theta}(x)$ はいい感じにグニャグニャしてほしいが、過剰にグニャグニャしないでほしい.

予測関数に“硬さ”のようなものを入れたい.

ニューラルネットの変動を抑える

積分型 : Higher-Order Variation Regularization (HOVR; Okuno, arXiv:2308.02293v1)

$$C_{k,q}(f_\theta) := \int_{\Omega} \left| \frac{\partial^k f_\theta(x)}{\partial^k x} \right|^q dx.$$

例えば正規直交基底を用いた回帰関数 $f_\theta(x) = \sum_j \theta_j \phi_j(x)$ の場合には,

$$C_{k,2}(f_\theta) \sim \|\theta\|_2^2$$

なので, パラメータ正則化の一般化と思える.

- ▶ 導関数は autograd で計算可能.
- ▶ 流行のPhysics-Informed Neural Network (PINN) の亜種と思える⁴.

⁴PINN では固定の Collocation Points を使うことが多い.

つまりどうこと?

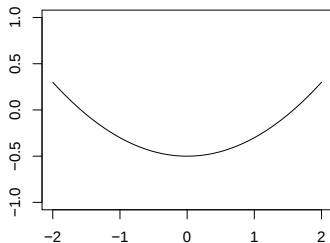


図: $C_{2,2}(f) \approx 0.64$

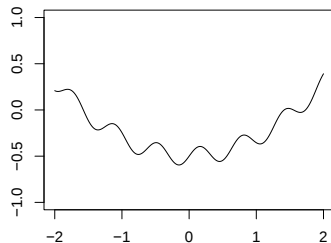


図: $C_{2,2}(f) \approx 197$

- ▶ HOVRを入れるとニューラルネットの激しいグニャグニャを抑制できる.
- ▶ 線形/カーネル回帰モデルなどではパラメータ正則化に対応. $\|f\|_{\mathcal{H}}$ は昔から….

提案法 (Okuno and Yagishita, arXiv:2308.02293, in revision)

- ▶ 正則化により過剰なグニャグニャを防ぎ，ロバスト損失で外れ値を無視する．

最小化する損失関数：

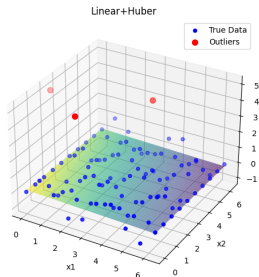
$$L(\theta) = \underbrace{\frac{1}{2} T_h(r(\theta))}_{\text{ロバスト損失}} + \underbrace{\lambda \cdot C_{k,q}(f_\theta)}_{\text{関数を硬くする正則化}}$$

- ▶ 破局点の理論解析などができる (省略)
- ▶ $C_{k,q}(f_\theta)$ は積分形だが，効率的なアルゴリズム⁵が作れる (省略)

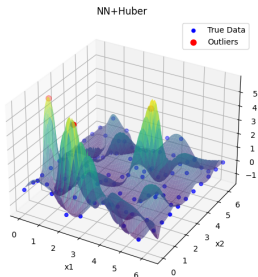
⁵Stochastic Gradient-Supergradient Descent (SGSD) は収束も証明できる．

というわけで最初の図に戻る.

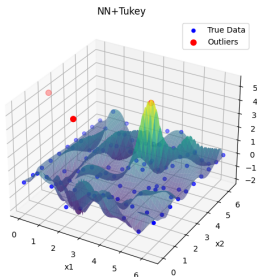
- ▶ (入力)2次元 - 100 - 100 - 100 - (出力)1次元の多層パーセプトロン.
- ▶ パラメータ数2万超, 外れ値3%混入.



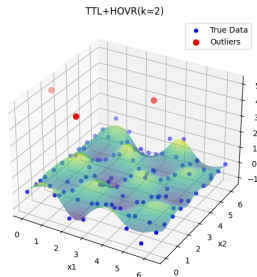
(a) 線形モデル+Huber



(b) NN+Huber



(c) NN+Tukey



(d) 提案法

図: $f_*(x) = \sin(2x_1) \cos(2x_2)$.

統計・機械学習に関する質問・依頼など、お気軽にご連絡ください。

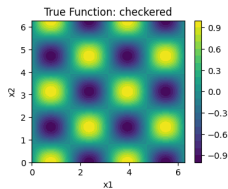
okuno@ism.ac.jp



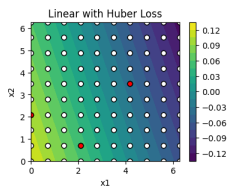
(今日の資料) <https://okuno.net/slides/2025-10-RIMS.pdf>

応用2の追加実験

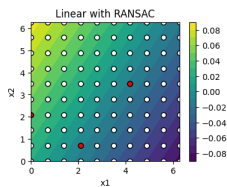
やってみた 1 (checked)



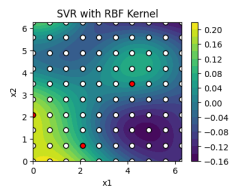
(a) Groundtruth



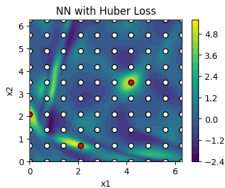
(b) Linear+Huber



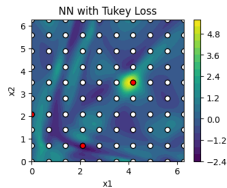
(c) Linear+RANSAC



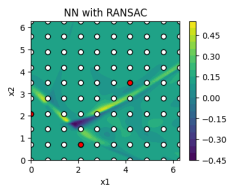
(d) SVR+RBF



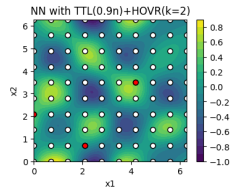
(e) NN+Huber



(f) NN+Tukey

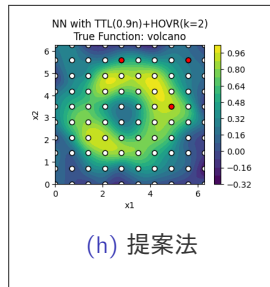
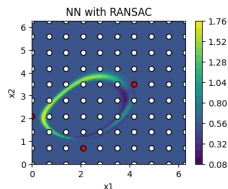
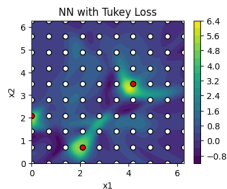
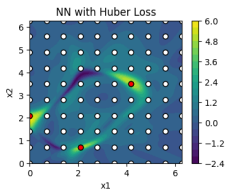
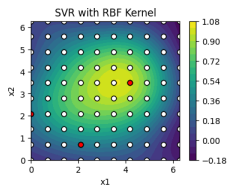
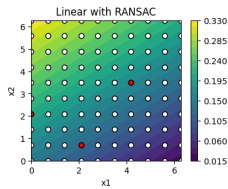
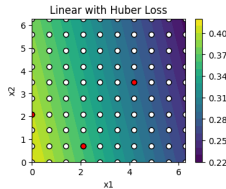
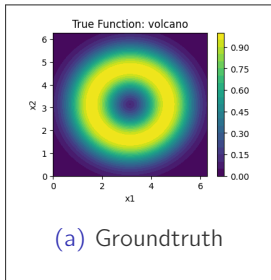


(g) NN+RANSAC

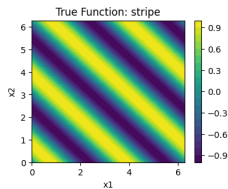


(h) 提案法

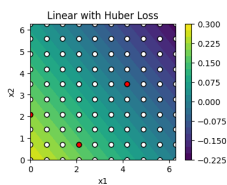
やってみた 2 (volcano)



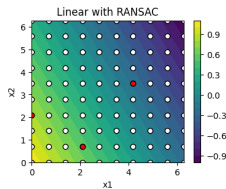
やってみた 3 (stripe)



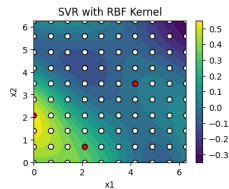
(a) Groundtruth



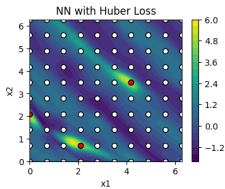
(b) Linear+Huber



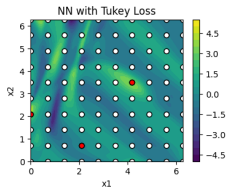
(c) Linear+RANSAC



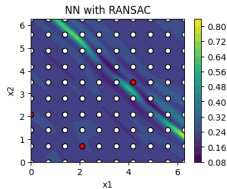
(d) SVR+RBF



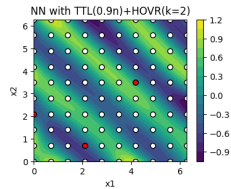
(e) NN+Huber



(f) NN+Tukey



(g) NN+RANSAC



(h) 提案法

やってみた 4 (plane)

