Nearest Neighbour Based Estimates of Gradients: Sharp Nonasymptotic Bounds and Applications

Guillaume Ausset[1,2], Stéphan Clémençon[1], François Portier[1]

[1] Télécom ParisTech, LTCI, Université Paris Saclay [2] BNP Paribas

Local Linear Estimation

We want to estimate the gradient of a function f, whose gradient is supposed sparse. We suppose that we are able to gather evaluations $Y = f(X)$ of the function of interest.

Local Linear Estimation of the Gradient

If we locally approximate the function by its Taylor expansion, we can estimate the gradient by solving

$$(\tilde{m}_k(x), \tilde{\beta}_k(x)) \in \underset{(m,\beta)\in\mathbb{R}^{D+1}}{\arg\min} \sum_{i:X_i\in \mathtt{KNN}(x)} (Y_i - m - \beta^\intercal(X_i - x))^2 + \lambda\|\beta\|_1$$

Theorem 1.

Let $n \geq 1$ and $k \geq 1$ such that $\overline{\tau}_k \leq \tau_0$. Let $\delta \in (0,1)$ and set $\lambda = \overline{\tau}_k(\sqrt{2\sigma^2\log(16D/\delta)/k} + L_2\overline{\tau}_k^2)$. Then, we have with probability larger than $1-\delta$,

$$\|\tilde{\beta}_k(x) - \beta(x)\|_2 \leq (24)^2\sqrt{\#\mathcal{S}_x}\left(\overline{\tau}_k^{-1}\sqrt{\frac{2\sigma^2\log(16D/\delta)}{k}} + L_2\overline{\tau}_k\right),$$

as soon as $C_1\#\mathcal{S}_x\log(Dn/\delta) \leq k \leq C_2 n$, $\overline{\tau}_k^2 \leq (b_f^2/(C_3\#\mathcal{S}_x L^2) \wedge \tau_0^2)$, where C_1, C_2 and C_3 are universal constants.

Our bounds make use of the *sparsity* of ∇f and only require a neighbourhood consisting of the K nearest points.

The use of a KNN neighbourhood gives us:

- Robustness to the data
- Ease of calibration
- Possibility to reuse past computations

Variable Selection

Require: (X, Y): training set, `Node`: indexes of points in the node

1: $\nabla m(X_i) \leftarrow$ `estimated gradient at` $X_i, \forall i \in$ `Node using` (1)
2: $\omega \leftarrow \sum_{i\in\mathtt{Node}} |\nabla m(X_i)|$
3: $K \leftarrow$ `sample` $\sqrt{D}$ `dimensions in` $\{1,\dots,d\}$ `with probability weights` $\propto \omega$
4: $k, c \leftarrow$ `best threshold` c `and dimension` k
5: **return** k, c

Algorithm 1: Node Splitting for Gradient Guided Trees

Variable Selection

	Description		Loss	
Dataset	n	D	RF	GGF
Wisconsin	569	30	0.0352	**0.0345**
Heart	303	13	0.128	**0.124**
Diamonds	53940	23	680033	**664265**
Gasoline	60	401	0.678	**0.512**
SDSS	10000	8	$0.872 \cdot 10^{-3}$	$\mathbf{0.776} \cdot 10^{-3}$

Table 1: Performance of the two random forest algorithms on a 50-folds cross validation.

Gradient Free Optimization

Require: x_0: initial guess, f: function $\mathbb{R}^D \to \mathbb{R}$, M: budget

1: $X \leftarrow X_1, \dots, X_M$ `with` $X_i \sim \mathcal{N}(x_0, \varepsilon \times I_D)$
2: $Y \leftarrow f(X) := f(X_1), \dots, f(X_M)$
3: **while** `not StoppingCondition` **do**
4: $\quad m, \Delta \leftarrow$ `estimated gradient at` x `w.r.t` X, Y `using` (1)
5: $\quad X \leftarrow X, X_1, \dots, X_M$ `with` $X_i \sim \mathcal{N}(\mathtt{GradientStep}(x, \Delta), \varepsilon \times I_D)$
6: $\quad Y \leftarrow f(X)$
7: $\quad x \leftarrow \arg\min_{X_i}\{f(X_i)\}$
8: **return** x

Algorithm 2: Estimated Gradient Descent

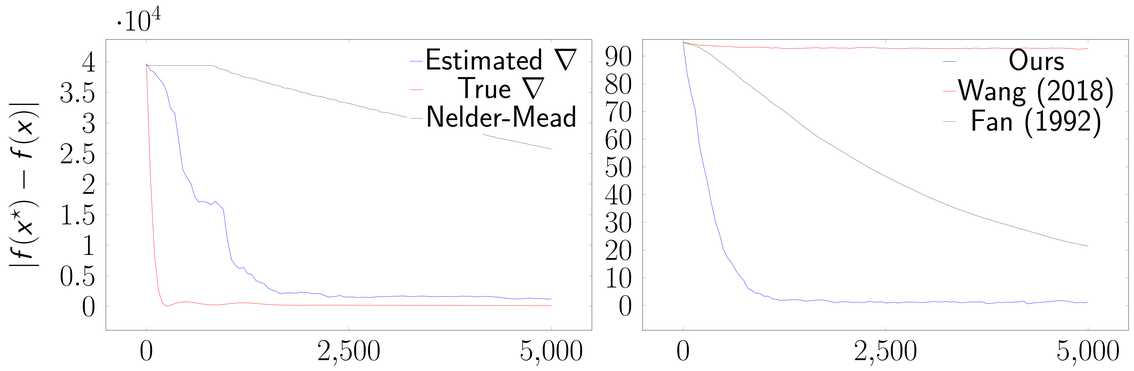

Figure 1: Gradient Descent on the sparse noisy Rosenbrock function for $d = 100$.

Work financed by BNP Paribas

firstname.lastname@telecom-paristech.fr