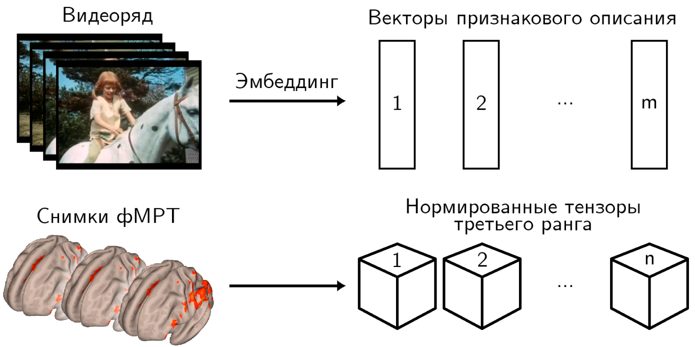
Методы оптимизации линейных задач большой размерности

Дорин Даниил, Киселев Никита

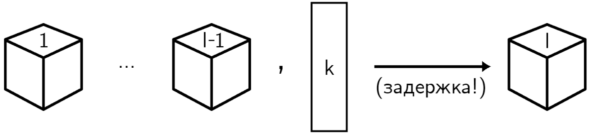
Московский физико-технический институт

12 мая 2023 г.

Мотивация



Восстановление следующего снимка фМРТ
по одному изображению и предшествующим снимкам



Пусть $\mathcal{P}$ — видеоряд, ν — частота изображений в видеоряде, t — продолжительность видеоряда:

$$\mathcal{P} = (\boldsymbol{p}^1, \ldots, \boldsymbol{p}^{\nu \cdot t}),\ \boldsymbol{p}^l \in \mathbb{R}^{W_{\boldsymbol{p}} \times H_{\boldsymbol{p}} \times C_{\boldsymbol{p}}}$$

где $W_{\boldsymbol{p}}$ — ширина изображения, $H_{\boldsymbol{p}}$ — высота изображения и $C_{\boldsymbol{p}}$ — число каналов.

$\mathcal{S}$ — последовательность фМРТ снимков некоторого испытуемого, μ — частота снимков фМРТ:

$$\mathcal{S} = (\boldsymbol{s}^1, \ldots, \boldsymbol{s}^{\mu \cdot t}),\ \boldsymbol{s}^l \in \mathbb{R}^{X_{\boldsymbol{s}} \times Y_{\boldsymbol{s}} \times Z_{\boldsymbol{s}}}$$

где $X_{\boldsymbol{s}}$, $Y_{\boldsymbol{s}}$, $Z_{\boldsymbol{s}}$ — размерности тензора снимка фМРТ.

Считаем, что известно несколько предыдущих измерений фМРТ $\mathcal{S}_0$ того же испытуемого. Необходимо построить отображение $\boldsymbol{f}$:

$$\boldsymbol{f}(\boldsymbol{p}^1, \ldots, \boldsymbol{p}^{k_i - \nu \cdot \Delta t}, \mathcal{S}_0) = \boldsymbol{s}^i,\ \mathcal{S}_0 = (\boldsymbol{s}_0^1, \ldots, \boldsymbol{s}_0^{i-1})$$

которое учитывает задержку Δt.

$$k_i = \frac{\nu \cdot i}{\mu}$$

Предположение марковости: $\boldsymbol{s}^i$ зависит лишь от $\boldsymbol{s}^{i-1}$ и $\boldsymbol{p}^{k_i-\nu\cdot\Delta t}$.

$$\text{Отображение}: \ \boldsymbol{f}(\boldsymbol{p}^{k_i-\nu\cdot\Delta t}) = \boldsymbol{\delta}^i, \ i = 2, \dots, L_{\Delta t},$$

$$\boldsymbol{f} = \boldsymbol{g} \circ \boldsymbol{v},$$

$$\boldsymbol{v}: \ \mathbb{R}^{W_{\boldsymbol{p}} \times H_{\boldsymbol{p}} \times C_{\boldsymbol{p}}} \to \mathbb{R}^d, \ \boldsymbol{g}: \ \mathbb{R}^d \to \mathbb{R}^{X_{\boldsymbol{s}} \times Y_{\boldsymbol{s}} \times Z_{\boldsymbol{s}}}.$$

- $L_{\Delta t}$ число пар снимок-изображение из обучающей выборки
- $\boldsymbol{\delta}^i = \boldsymbol{s}^i - \boldsymbol{s}^{i-1}$, $\boldsymbol{s}^k \in \mathbb{R}^{X_{\boldsymbol{s}} \times Y_{\boldsymbol{s}} \times Z_{\boldsymbol{s}}}$
- δ_{ijk} — компонента тензора $\boldsymbol{\delta}$.
- $\boldsymbol{x} = \boldsymbol{v}(\boldsymbol{p})$, $\boldsymbol{x} = [x_1, \dots, x_d]^T$ — вектор признаков изображения
- $\boldsymbol{w}_{ijk} = [w_1^{ijk}, \dots, w_d^{ijk}]^T$ — вектор весов элемента тензора δ_{ijk}

Для восстановления разности значений в каждом вокселе используется линейная модель

$$g_{ijk}(\boldsymbol{x}, \boldsymbol{w}_{ijk}) = \langle \boldsymbol{x}, \boldsymbol{w}_{ijk} \rangle.$$

Для каждого вокселя в снимке задана обучающая выборка. N — число объектов тренировочной выборки. Функция потерь:

$$\mathcal{L}_{ijk}(\boldsymbol{w}_{ijk}, \Delta t, \alpha) = \frac{1}{2} \sum_{l=2}^{N+1} \left(g_{ijk}(\boldsymbol{x}^l, \boldsymbol{w}_{ijk}) - \delta_{ijk}^l \right)^2 + \frac{\alpha}{2} \|\boldsymbol{w}_{ijk}\|_2^2.$$

Требуется минимизировать функцию потерь $\mathcal{L}_{ijk}(\mathbf{w}_{ijk}, \Delta t, \alpha)$ при фиксированных гиперпараметрах Δt и α:

$$\hat{\mathbf{w}}_{ijk} = \arg\min_{\mathbf{w}_{ijk}} \mathcal{L}_{ijk}(\mathbf{w}_{ijk}, \Delta t, \alpha).$$

Определим матрицу объектов тренировочной выборки

$$\mathbf{X} = [\mathbf{x}^{2^T}, \dots, \mathbf{x}^{N+1^T}]^T = [x_j^i] \in \mathbb{R}^{N \times d}$$

и вектор, компонентами которого являются разности значений одного и того же вокселя в снимках тренировочной выборки,

$$\mathbf{\Delta}_{ijk} = [\delta_{ijk}^2, \dots, \delta_{ijk}^{N+1}]^T \in \mathbb{R}^N.$$

Тогда решение МНК можно записать в виде:

$$\hat{\mathbf{w}}_{ijk} = (\mathbf{X}^T\mathbf{X} + \alpha \mathrm{I})^{-1}\mathbf{X}^T\mathbf{\Delta}_{ijk}.$$

Параметры выборки

1. $L_{\Delta t} = 641 - \mu \cdot \Delta t$, $\nu = 25 \text{ c}^{-1}$, $\mu = 1.64 \text{ c}^{-1}$
2. $\boldsymbol{p} \in \mathbb{R}^{640 \times 480 \times 3}$, $\boldsymbol{s} \in \mathbb{R}^{40 \times 64 \times 64}$

При предобработке картинок остаточной CNN так или иначе теряется информация. Целью данной работы является сравнение различных методов решения задач линейной регрессии большой размерности. Предлагается исследовать различные методы ускоренного градиентного спуска, адаптивные методы, покомпонентные методы, SGD и Mini-Batch SGD. В рассматриваемой выборке размерность вектора неизвестных составляет порядка 10^6.

Класс рассматриваемых задач

В качестве функции потерь по-прежнему используем среднеквадратичное отклонение, но уже без регуляризации. В матричном виде задача имеет следующий вид:

$$f(w) = \|Xw - y\|_2^2, \ X \in \mathbb{R}^{447 \times 921600}$$

$$f(w) \to \min$$

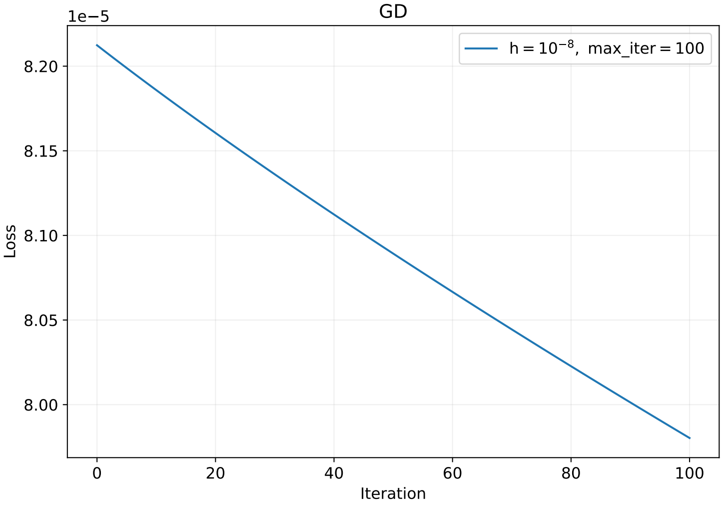
$$\nabla f(w) = 2X^\top (Xw - y)$$

$$\nabla^2 f(w) = 2X^\top X, \ X^\top X \in \mathbb{R}^{921600 \times 921600}$$

В дальнейшем считаем, что задача является выпуклой с вырожденной матрицей X, Липшиц-гладкой с некоторой неизвестной константой L, которую не предоставляется возможным посчитать в силу большой размерности матрицы.

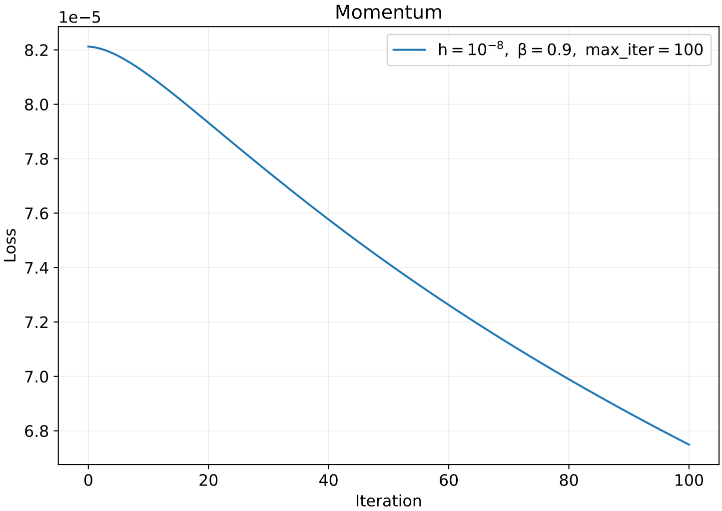
Градиентный спуск с постоянным шагом

$$w_{k+1} = w_k - h\nabla f(w_k)$$

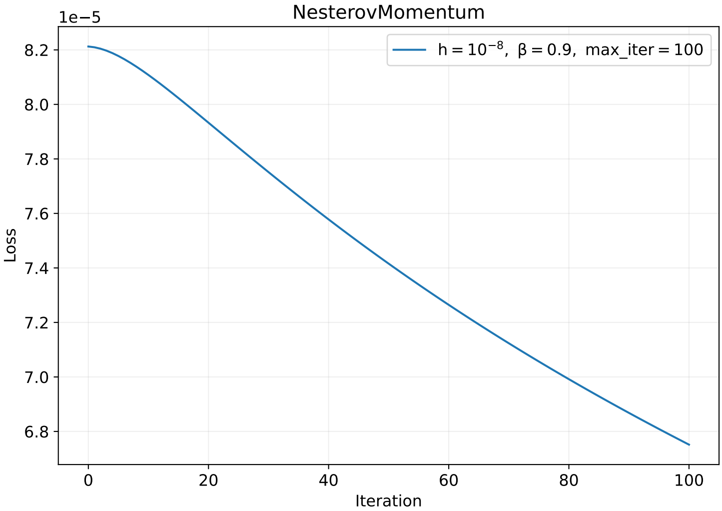


$$v_{k+1} = \beta v_k - h\nabla f(w_k)$$

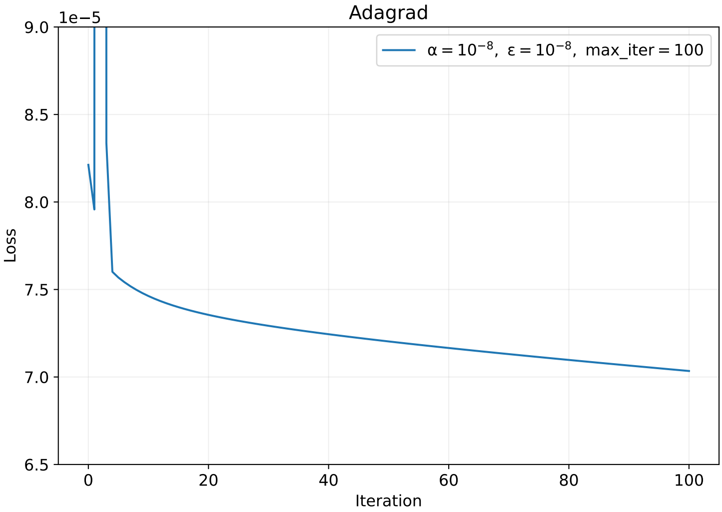
$$w_{k+1} = w_k + v_{k+1}$$



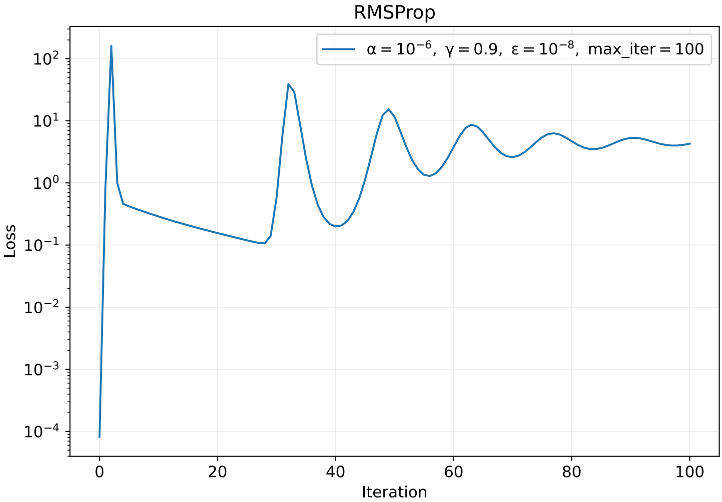
$$v_{k+1} = \beta v_k - h \nabla f(w_k + \beta v_k)$$
$$w_{k+1} = w_k + v_{k+1}$$



$$G_{k+1} = G_k + [\nabla f(w_k)]^2$$
$$w_{k+1} = w_k - \frac{\alpha}{\sqrt{G_{k+1} + \varepsilon}} \nabla f(w_k)$$



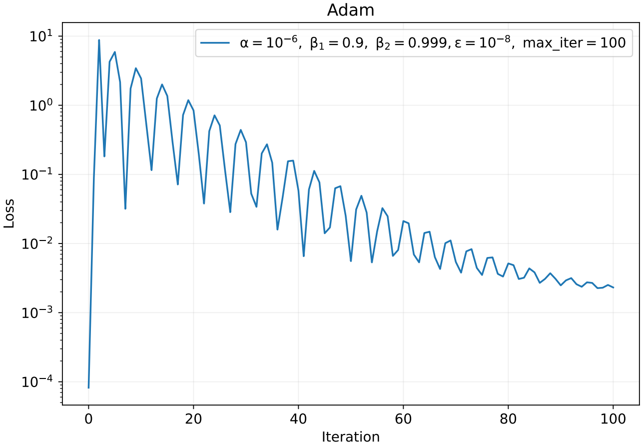
$$G_{k+1} = \gamma G_k + (1-\gamma)[\nabla f(w_k)]^2$$
$$w_{k+1} = w_k - \frac{\alpha}{\sqrt{G_{k+1}+\varepsilon}} \nabla f(w_k)$$



$$v_{k+1} = \beta_1 v_k + (1 - \beta_1)\nabla f(w_k)$$

$$G_{k+1} = \beta_2 G_k + (1 - \beta_2)[\nabla f(w_k)]^2$$

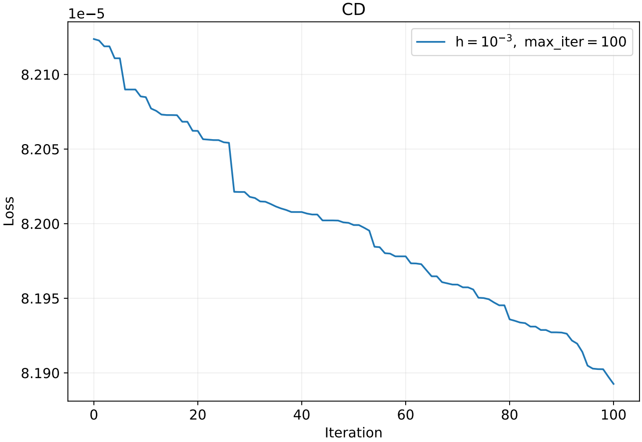
$$w_{k+1} = w_k - \frac{\alpha}{\sqrt{G_{k+1} + \varepsilon}} v_{k+1}$$



$$\text{Sample } j_k \in \{1, \dots, 921600\}$$

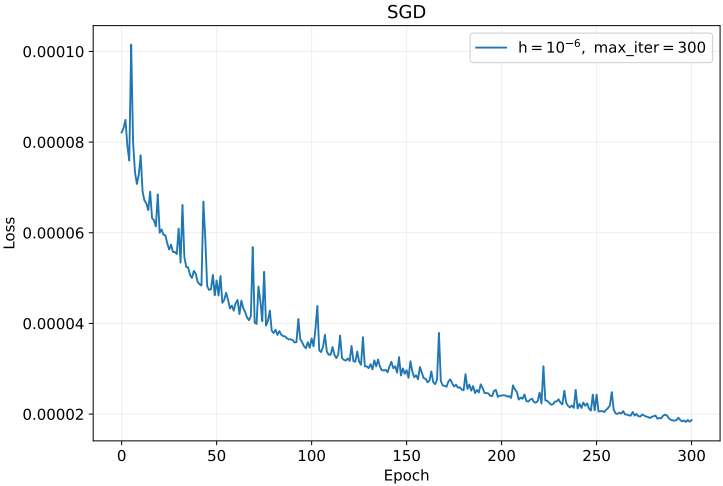
$$g_k = 2X_{j_k}^\top (Xw - y)$$

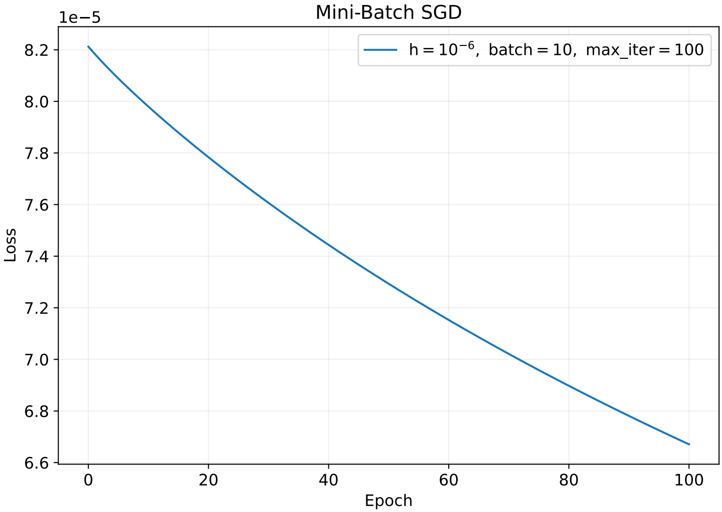
$$w_{j_k}^{k+1} = w_{j_k}^k - hg_k$$

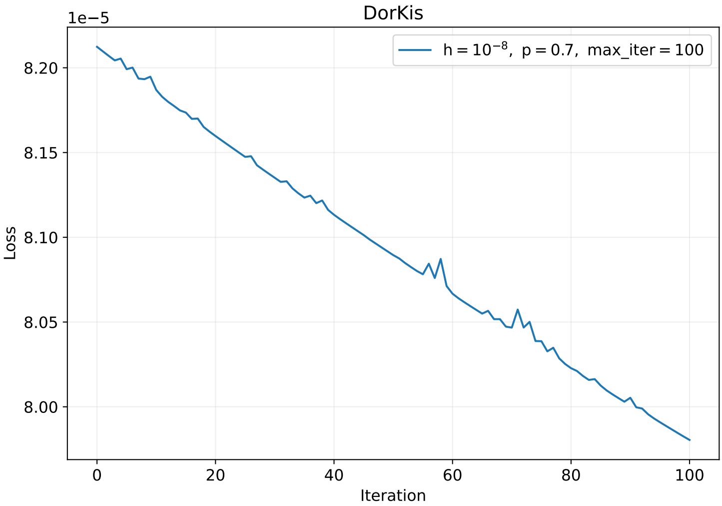


$$\text{Sample } i_k \in \{1, \ldots, 447\}$$

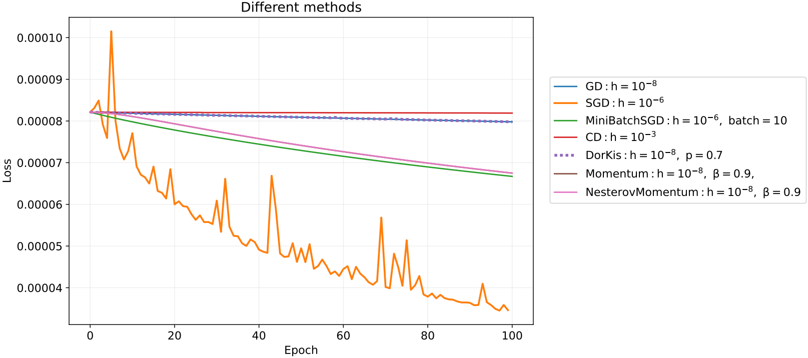
$$w_{k+1} = w_k - h \nabla_{i_k} f(w_k)$$



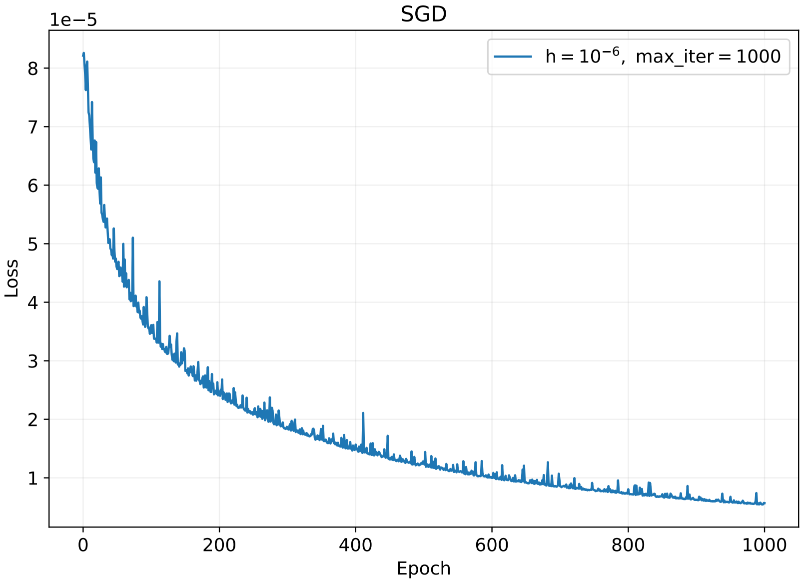

Mini-Batch SGD
1e−5
8.2
8.0
7.8
7.6
7.4
7.2
7.0
6.8
6.6
Loss
0
20
40
60
80
100
Epoch
$h = 10^{-6}$, batch = 10, max_iter = 100


DorKis
1e−5
8.20
8.15
8.10
8.05
8.00
Loss
0
20
40
60
80
100
Iteration
h = 10^−8, p = 0.7, max_iter = 100

Сравнение методов



Лучший метод



Спасибо за внимание!