Group and Shuffle: Efficient Structured Orthogonal Parametrization

Mikhail Gorbunov [1] Nikolay Yudin [1] Vera Soboleva [2, 1] Aibek Alanov [2, 1]
Alexey Naumov [1, 3] Maxim Rakhuba [1]

[1]HSE University [2]AIRI [3]Steklov Mathematical Institute RAS

1. Introduction

We propose a new low-dimensional subset of orthogonal matrices and a way to efficiently parametrize it. In particular, our contributions are:

- A new class of $\mathcal{GS}$-matrices that generalizes multiple previous structures.
- An optimal way to form a dense matrix in the $\mathcal{GS}$-class.
- Efficient structured orthogonal parametrization of $\mathcal{GS}$-matrices.
- Comparison of our class and existing methods in orthogonal fine-tuning (LMs and diffusion models) and orthogonal convolutional architectures.

2. Orthogonal Fine-Tuning

Orthogonal Fine-Tuning framework modifies forward pass of pre-trained linear layers:

$$\boldsymbol{y} = (\boldsymbol{W^0})^\top \boldsymbol{x} \quad \rightarrow \quad \boldsymbol{y} = (\boldsymbol{Q}\boldsymbol{W^0})^\top \boldsymbol{x}; \qquad \boldsymbol{Q} \text{ – orthogonal matrix.}$$

- **OFT** method [Qiu et al., 2023] uses block-diagonal structure for $\boldsymbol{Q}$, parametrizing it as
$$\boldsymbol{Q} = \mathbf{diag}(\boldsymbol{Q_1}, \boldsymbol{Q_2}, \dots, \boldsymbol{Q_r}),$$
where $\boldsymbol{Q_i} \in \mathbb{R}^{b\times b}$ are small orthogonal matrices.
Problem: restrictive structure.
- **BOFT** method [Liu et al., 2024] parametrizes $\boldsymbol{Q}$ as a product of several orthogonal sparse matrices, aiming to form a dense orthogonal matrix:
$$\boldsymbol{Q} = \boldsymbol{B_m}\boldsymbol{B_{m-1}}\dots\boldsymbol{B_1}.$$
$\boldsymbol{B_i}$ is (up to permutations) a block-diagonal matrix with $\boldsymbol{r}$ orthogonal $\boldsymbol{b}\times\boldsymbol{b}$ blocks.
Problem: requires computing a product of multiple matrices. For example, for $\boldsymbol{Q}$ of the size $\mathbf{1024}\times\mathbf{1024}$ and for $\boldsymbol{r}=\mathbf{32}$, we have $\boldsymbol{m}=\mathbf{6}$.

4. Orthogonal $\mathcal{GS}$-matrices

We can achieve orthogonality through enforcing it for every block of block-diagonal matrix $\boldsymbol{B_i}$. This is theoretically justified, as shown by **Theorem 2**.

Theorem 2

If each block of every $\boldsymbol{B_i}$ in $\mathcal{GS}(\boldsymbol{P}_{m+1}, \dots, \boldsymbol{P}_1)$ is orthogonal then $\mathcal{GS}(\boldsymbol{P}_{m+1}, \dots, \boldsymbol{P}_1)$ is orthogonal matrix. In case $\boldsymbol{m}=\mathbf{2}$, any orthogonal matrix from $\mathcal{GS}(\boldsymbol{P}_3, \boldsymbol{P}_2, \boldsymbol{P}_1)$ admits $\boldsymbol{P}_3(\boldsymbol{B}_2\boldsymbol{P}_2\boldsymbol{B}_1)\boldsymbol{P}_1$ representation with the matrices $\boldsymbol{B}_1, \boldsymbol{B}_2$ consisting of orthogonal blocks.

Remark: This theorem allows us to maintain orthogonality of smaller blocks instead of preserving orthogonality of whole matrix.

Parametrizing blocks: We mostly use *Cayley transform* to maintain orthogonality of each smaller block in $\boldsymbol{B_i}$: for any skew-symmetric matrix $\boldsymbol{K}^\top = -\boldsymbol{K}$, we know that $\boldsymbol{Q}$, given by

$$\boldsymbol{Q} = (\boldsymbol{I}+\boldsymbol{K})(\boldsymbol{I}-\boldsymbol{K})^{-1}$$

is orthogonal matrix. Alternatively, one can enforce orthogonality with the help of matrix exponent: for any skew-symmetric $\boldsymbol{K}^\top = -\boldsymbol{K}$, we have

$$\boldsymbol{Q} = \exp(\boldsymbol{K}), \quad \exp(\boldsymbol{K}) = \sum_{n=0}^{\infty}\frac{\boldsymbol{K}^n}{\boldsymbol{n!}},$$

where $\boldsymbol{Q}$ is orthogonal.

3. $\mathcal{GS}$-matrices

Definition 1

$\boldsymbol{A}$ is said to be in $\mathcal{GS}(\boldsymbol{P}_{m+1}, \dots, \boldsymbol{P}_1)$ if

$$\boldsymbol{A} = \boldsymbol{P}_{m+1}\boldsymbol{B}_m\boldsymbol{P}_m\dots\boldsymbol{B}_1\boldsymbol{P}_1,$$

where each matrix $\boldsymbol{B_i}$ is a block-diagonal matrix with $\boldsymbol{k_i}$ blocks of size $\boldsymbol{b_i^1}\times\boldsymbol{b_i^2}$, matrices $\boldsymbol{P_i}$ are permutation matrices and $\boldsymbol{b_i^1}\cdot\boldsymbol{k_i} = \boldsymbol{b_{i+1}^2}\cdot\boldsymbol{k_{i+1}}$.

Remark: $\mathcal{GS}$-class contains OFT, BOFT, order-$\boldsymbol{p}$ Monarch matrices and more.

Crucial question

How to choose $\boldsymbol{P_i}$ to minimize $\boldsymbol{m}$ and ensure that $\boldsymbol{A}$ is dense?

Answer

Make $\boldsymbol{P_i}$ a *perfect shuffle* matrix $\boldsymbol{P}_{(r,br)}$! To visualize $\boldsymbol{y} = \boldsymbol{P}_{(r,br)}\boldsymbol{x}$ imagine splitting a deck (vector $\boldsymbol{x}$) into $\boldsymbol{b}$ piles and taking sequentially upper card from each pile to form $\boldsymbol{y}$.

The answer in a more formal way:

Theorem 1

Let $\boldsymbol{k_i} = \boldsymbol{r}, \boldsymbol{b_i^1} = \boldsymbol{b_i^2} = \boldsymbol{b}$. Then using $\boldsymbol{m} = \mathbf{1} + \lceil \log_b(\boldsymbol{r}) \rceil$ is sufficient for the class $\mathcal{GS}(\boldsymbol{P_L}, \boldsymbol{P}_{(r,br)}, \dots, \boldsymbol{P}_{(r,br)}, \boldsymbol{P_R})$ to form a dense matrix for any $\boldsymbol{P_L}, \boldsymbol{P_R}$. Moreover, the choice of $\boldsymbol{P}_2 = \dots = \boldsymbol{P_m} = \boldsymbol{P}_{(r,br)}$ is optimal in the sense that all matrices from $\mathcal{GS}(\boldsymbol{P}_{m+1}, \dots, \boldsymbol{P}_1)$ contain zero blocks for any integer $\boldsymbol{m} < \mathbf{1} + \lceil \log_b(\boldsymbol{r}) \rceil$ and any permutations $\boldsymbol{P}_1, \dots, \boldsymbol{P}_{m+1}$.

We also find that class $\mathcal{GS}(\boldsymbol{I}, \boldsymbol{P}, \boldsymbol{I})$ consists of block matrices with low-rank blocks, whose ranks are defined by permutation matrix $\boldsymbol{P}$.

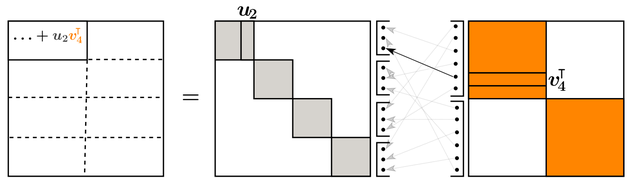

Figure 1. Illustration of block low-rank interpretation of $\mathcal{GS}(\boldsymbol{I}, \boldsymbol{P}, \boldsymbol{I})$ matrices.

Applications

- **NLP** Learnable orthogonal matrices for PEFT methods prevents training instabilities and overfitting that alternative methods like LoRA suffer from.
- **Diffusion models** Orthogonal fine-tuning of diffusion models helps to impose specific properties to a model without losing quality of generation.
- **Convolutional architectures** Equipping convolution operator with particular properties allows to obtain convolution with Lipschitz constant equal to 1.

Bibliography

Weiyang Liu, Zeju Qiu, Yao Feng, Yuliang Xiu, Yuxuan Xue, Longhui Yu, Haiwen Feng, Zhen Liu, Juyeon Heo, Songyou Peng, Yandong Wen, Michael J. Black, Adrian Weller, and Bernhard Schölkopf. Parameter-efficient orthogonal finetuning via butterfly factorization. In *The Twelfth International Conference on Learning Representations*, 2024. URL **https://openreview.net/forum?id=7NzgkEdGyr**.

Zeju Qiu, Weiyang Liu, Haiwen Feng, Yuxuan Xue, Yao Feng, Zhen Liu, Dan Zhang, Adrian Weller, and Bernhard Schölkopf. Controlling text-to-image diffusion by orthogonal finetuning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL **https://openreview.net/forum?id=K30wTdIIYc**.

Sahil Singla and Soheil Feizi. Skew orthogonal convolutions. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9756–9766. PMLR, 18–24 Jul 2021. URL **https://proceedings.mlr.press/v139/singla21a.html**.

Application 1: NLP

We utilize the pipeline of orthogonal fine-tuning parametrizing multiplicative matrix $\boldsymbol{Q}$ with $\mathcal{GS}(\boldsymbol{P}_{(r,br)}^\top, \boldsymbol{P}_{(r,br)}, \boldsymbol{I})$-matrices:

$$\boldsymbol{Q} = \boldsymbol{P}_{(r,br)}^\top \boldsymbol{B_2} \boldsymbol{P}_{(r,br)} \boldsymbol{B_1}.$$

Method	# Params	MNLI	SST-2	CoLA	QQP	QNLI	RTE	MRPC	STS-B	ALL
FT	125M	87.62	94.38	61.97	**91.5**	**93.06**	80.14	88.97	**90.91**	86.07
LoRA$_{r=8}$	1.33M	**87.82**	**95.07**	64.02	90.97	92.81	**81.95**	88.73	90.84	86.53
OFT$_{b=16}$	1.41M	87.21	**95.07**	64.37	90.6	92.48	79.78	89.95	90.71	86.27
BOFT$^{m=2}_{b=8}$	1.42M	87.14	94.38	62.57	90.48	92.39	80.14	88.97	90.67	85.84
GSOFT$_{b=8}$	1.42M	87.16	**95.06**	**65.3**	90.46	92.46	**81.95**	**90.2**	90.76	**86.67**

Table 1. Results on GLUE benchmark with RoBERTa-base model. We report Pearson correlation for STS-B, Matthew's correlation for CoLA and accuracy for other tasks.

Application 2: Subject-driven generation

For pre-trained diffusion models we investigate an approach that multiplies weight matrices from both sides with $\boldsymbol{Q_U}, \boldsymbol{Q_V}$ parametrized as orthogonal $\mathcal{GS}$-matrices:

$$\boldsymbol{y} = (\boldsymbol{W^0})^\top \boldsymbol{x} \quad \rightarrow \quad \boldsymbol{y} = (\boldsymbol{Q_U}\boldsymbol{W^0}\boldsymbol{Q_V})^\top \boldsymbol{x}$$

Compared to one-sided adaptation, this modification allows to adapt both left and right singular vectors of $\boldsymbol{W^0}$, improving method's flexibility.

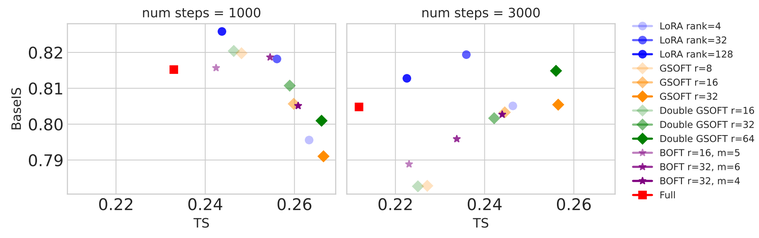

Figure 2. Image and text similarity visualization for different methods on subject-driven generation.

Application 3: Orthogonal Convolutions

We can also impose orthogonality to the convolution layer using skew-symmetric filters. Such constraint helps maintaining Lipschitz constant equal to 1 during training.

$$\boldsymbol{Y} = \texttt{ChShuffle}_2(\boldsymbol{L}^{(2)}_{grouped} \star_e (\texttt{ChShuffle}_1(\boldsymbol{L}^{(1)}_{grouped} \star_e \boldsymbol{X})))$$

$$\boldsymbol{L} \star_e \boldsymbol{X} = \boldsymbol{X} + \frac{\boldsymbol{L}\star\boldsymbol{X}}{\mathbf{1!}} + \dots + \frac{\boldsymbol{L}\star^n\boldsymbol{X}}{\boldsymbol{n!}} + \dots, \quad \star^i - \text{convolution applied i times}$$

Table 2. Results of LipConvnet-15 on CIFAR-100. $(\boldsymbol{a}, \boldsymbol{b})$ in "Groups" denotes that we have two grouped exponential convolutions (first with $\boldsymbol{kernel\_size}=\mathbf{3}$, second with $\boldsymbol{kernel\_size}=\mathbf{1}$). If $\boldsymbol{b}=$ "-", we only use one $\mathcal{GS}$ orthogonal convolution. We use `ChShuffle` before each grouped convolution.

Conv. Layer	# Params	Groups	Speedup	Accuracy	Robust Accuracy
SOC [Singla and Feizi, 2021]	24.1M	-	1	43.15%	29.18%
GS-SOC (Ours)	**6.81M**	(4, -)	**1.64**	**43.48%**	**29.26%**
GS-SOC (Ours)	**8.91M**	(4, 1)	1.21	**43.42%**	**29.56%**
GS-SOC (Ours)	**7.86M**	(4, 2)	1.22	42.86%	28.98%
GS-SOC (Ours)	**7.3M**	(4, 4)	1.23	42.75%	28.7%