Scalable Second-Order Optimization Algorithms for Minimizing Low-Rank Functions

Edward Tansley, Coralia Cartis, Zhen Shao

Abstract

The work we present is featured in the papers [1] and [2][1]. We focus on hidden structures contained within optimization problems and algorithms that can exploit these. Low-rank functions [3] are those that vary only in some (low-dimensional) linear subspace. Specifically, there exists a subspace $\mathcal{T}$ of dimension r such that $f(\boldsymbol{x}_\top + \boldsymbol{x}_\perp) = f(\boldsymbol{x}_\top)$ for all $\boldsymbol{x}_\top \in \mathcal{T}$ and $\boldsymbol{x}_\perp \in \mathcal{T}^\perp$, for some $r \leq d$. Low-rank functions arise when tuning (over)parametrized models and processes, such as in hyperparameter optimization for neural networks, heuristic algorithms for combinatorial optimization problems and physical simulation problems including climate modelling. We seek to exploit the structure of low-rank functions to develop scalable second-order optimization algorithms for high-dimensional problems.

Second-order optimization algorithms for the unconstrained optimization problem $\min_{x\in\mathbb{R}^d} f(\boldsymbol{x})$, where $f : \mathbb{R}^d \to \mathbb{R}$ is a sufficiently smooth, bounded-below function, use gradient and curvature information to determine iterates and thus, may have faster convergence than first-order algorithms that only rely on gradient information. However, for high-dimensional problems, when d is large, the computational cost of these methods can be a barrier to their use in practice. We are concerned with the task of scaling up second-order optimization algorithms so that they are a practical option for certain high-dimensional problems.

An alternative to linesearch and trust-region techniques, Adaptive Regularization framework using Cubics (ARC) [4] determines the change $\boldsymbol{s}_k$ between the current iterate $\boldsymbol{x}_k$ and $\boldsymbol{x}_{k+1}$ by (approximately) solving a cubically regularized local quadratic model:

$$\arg\min_{\boldsymbol{s}\in\mathbb{R}^d} m_k(\boldsymbol{s}) = f(\boldsymbol{x}_k) + \langle \nabla f(\boldsymbol{x}_k),\ \boldsymbol{s}\rangle + \frac{1}{2}\langle \boldsymbol{s},\ \nabla^2 f(\boldsymbol{x}_k)\boldsymbol{s}\rangle + \frac{\sigma_k}{3}||\boldsymbol{s}||_2^3,$$

where ∇f and $\nabla^2 f$ denote the gradient and Hessian of f and $\sigma_k > 0$ is the regularization parameter. Assuming Lipschitz continuity of the Hessian on the iterates' path, ARC requires at most $\mathcal{O}\left(\epsilon^{-3/2}\right)$ iterations and function and (first- and second-) derivative evaluations to attain an ϵ-approximate first-order critical point; this convergence rate is optimal over a large class of second-order methods.

R-ARC is a random subspace variant of ARC and restricts the step $\boldsymbol{s}_k$ to some $l-$dimensional subspace spanned by the rows of some randomly-drawn scaled Gaussian matrix $\boldsymbol{S}_k \in \mathbb{R}^{l\times d}$. That is, we have

$$\begin{aligned}&\arg\min_{\hat{\boldsymbol{s}}\in\mathbb{R}^l} \hat{m}_k(\hat{\boldsymbol{s}}) = f(\boldsymbol{x}_k) + \langle \nabla f(\boldsymbol{x}_k),\ \boldsymbol{S}_k^\top\hat{\boldsymbol{s}}\rangle + \frac{1}{2}\langle \boldsymbol{S}_k^\top\hat{\boldsymbol{s}},\ \nabla^2 f(\boldsymbol{x}_k)\boldsymbol{S}_k^\top\hat{\boldsymbol{s}}\rangle + \frac{\sigma_k}{3}\|\boldsymbol{S}_k^\top\hat{\boldsymbol{s}}\|_2^3\\ =&\arg\min_{\hat{\boldsymbol{s}}\in\mathbb{R}^l} \hat{m}_k(\hat{\boldsymbol{s}}) = f(\boldsymbol{x}_k) + \langle \hat{\nabla} f(\boldsymbol{x}_k),\ \hat{\boldsymbol{s}}\rangle + \frac{1}{2}\langle \hat{\boldsymbol{s}},\ \hat{\nabla}^2 f(\boldsymbol{x}_k)\hat{\boldsymbol{s}}\rangle + \frac{\sigma_k}{3}\langle \hat{\boldsymbol{s}},\ \boldsymbol{M}_k\hat{\boldsymbol{s}}\rangle^{\frac{3}{2}},\end{aligned}$$

where $\hat{\nabla} f(\boldsymbol{x}_k) = \boldsymbol{S}_k\nabla f(\boldsymbol{x}_k)$, $\hat{\nabla}^2 f(\boldsymbol{x}_k) = \boldsymbol{S}_k\nabla^2 f(\boldsymbol{x}_k)\boldsymbol{S}_k^\top$ and $\boldsymbol{M}_k = \boldsymbol{S}_k\boldsymbol{S}_k^\top$. Solving this subproblem is much less computationally expensive to solve than that of ARC (assuming we have $l \ll d$).

[1] https://arxiv.org/pdf/2501.03718 and https://arxiv.org/pdf/2501.09734 respectively

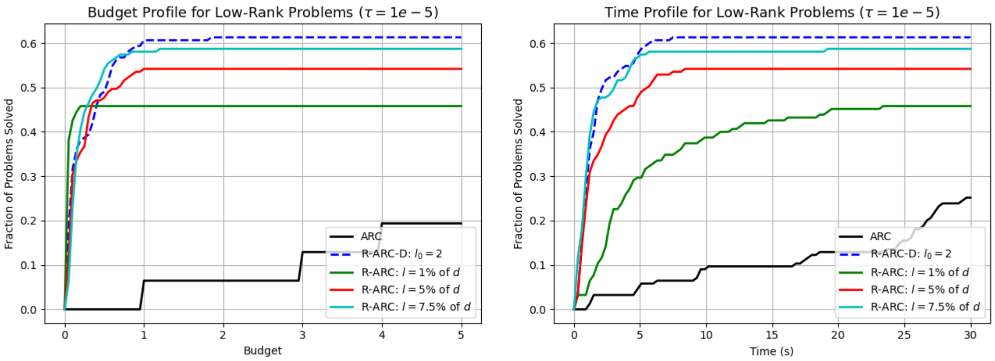

Figure 1: Data profiles of R-ARC-D compared to R-ARC and ARC on 31 low-rank CUTEst problems.

Additionally, we only require access to projected problem information, rather than full gradients and Hessians.

We prove that when R-ARC is applied to minimize a low-rank function of rank r, we need a subspace dimension of $l = \mathcal{O}(r)$ to achieve the same optimal $\mathcal{O}\left(\epsilon^{-3/2}\right)$ convergence rate as ARC, meaning R-ARC is scalable for high-dimensional problems with low-rank structure.

In R-ARC, l is fixed throughout the run of the algorithm and r may not be known a priori. In R-ARC-D, we allow an adaptive subspace dimension l_k which may vary through the run. We detail an update scheme that enables R-ARC-D to learn the rank of the function using projected problem information $\hat{r}_k := rank(\boldsymbol{S}_k \nabla^2 f(\boldsymbol{x}_k) \boldsymbol{S}_k^\top)$. We show that under this update scheme, R-ARC-D attains the optimal $\mathcal{O}\left(\epsilon^{-3/2}\right)$ convergence rate, regardless of the initial subspace dimension l_0.

We test the performance of R-ARC-D against R-ARC and ARC in numerical experiments on CUTEst problems as well as artificially created low-rank problems [1, 2]. An example is included in Figure 1 (taken from [1]), where R-ARC-D outperforms both R-ARC and ARC when applied to low-rank test problems of rank $r = 100$ and dimension $d = 1000$, even when the initial subspace dimension for R-ARC-D is set to be 2. This is in contrast to R-ARC which uses subspace dimensions of $l = 10,\ 50,\ 75$. Thus R-ARC-D is able to successfully learn and exploit the low-rank structure in the problems.

References

[1] Edward Tansley and Coralia Cartis. Scalable Second-Order Optimization Algorithms for Minimizing Low-rank Functions, January 2025. arXiv:2501.03718 [math].

[2] Coralia Cartis, Zhen Shao, and Edward Tansley. Random Subspace Cubic-Regularization Methods, with Applications to Low-Rank Functions, January 2025. arXiv:2501.09734 [math].

[3] Ziyu Wang, Frank Hutter, Masrour Zoghi, David Matheson, and Nando De Feitas. Bayesian Optimization in a Billion Dimensions via Random Embeddings. *Journal of Artificial Intelligence Research*, 55:361–387, February 2016.

[4] Coralia Cartis, Nicholas I. M. Gould, and Philippe L. Toint. Adaptive cubic regularisation methods for unconstrained optimization. Part I: motivation, convergence and numerical results. *Mathematical Programming*, 127(2):245–295, April 2011.