RESCIENCE C

Replication / ML Reproducibility Challenge 2022

[Re] Improve Single-Point Zeroth-Order Optimization Using High-Pass and Low-Pass Filters

Anonymous[1, ID]
[1]Anonymous Institution

Edited by
–

Reviewed by
–

Received
–

Published
–

DOI
–

Reproducibility Summary

Scope of Reproducibility – In this paper, the authors propose a **s**ingle-point **z**eroth-order **o**ptimization (SZO) algorithm called **H**igh/**L**ow-pass **F**ilter SZO (HLF-SZO), which they claim achieves (i) smaller variance compared to vanilla SZO, (ii) faster convergence compared to vanilla SZO and residual-feedback SZO, and (iii) comparable empirical performance to two-point ZO. The authors support (ii) by showing that HLF-SZO gives a better theoretical iteration complexity than vanilla SZO and residual-feedback SZO. Numerical experiments corroborate all claims on various synthetic convex/nonconvex problems.

Methodology – We reproduce two figures from the numerical experiments section of the paper to verify the aforementioned claims on several optimization problems. The authors do not provide any code, so all aspects of the reproduction (e.g., algorithms, figure generation) had to be done by interpreting the methods outlined in the paper. We also extend the results of the paper by evaluating HLF-SZO on the highly nonconvex problem of training a two-layer neural network. Additionally, we compare HLF-SZO to an SZO method derived from higher order filters using a similar approach to that in the paper. All code could be run on a laptop without GPU. Finally, we flesh out proof details of the convergence result for convex problems.

Results – For the figures selected, we were able to replicate the results almost identically, with little to no discernible visual differences between the original and reproduced figures. In particular, we verify the authors' claims that HLF-SZO converges in fewer iterations with less variance compared to vanilla SZO, converges quicker than residual-feedback SZO, yet is slightly slower than, yet comparable to, two-point ZO.

What was easy – Even though no code was provided, it was fairly easy to reimplement the algorithms as described in the paper due to their simplicity and brevity.

What was difficult – There was some initial confusion about what was being plotted in the figures due to overloaded notation in the body of the experimental section.

Communication with original authors – We contacted the original authors regarding details about the manual search procedure for hyperparameters of the method, to which they responded satisfactorily.

1 Introduction

This paper [1] considers solving a generic unconstrained optimization problem

$$\min_{x\in\mathbb{R}^d} f(x).$$

In applications such as black-box optimization and control theory, the derivatives of f may not be available or expensive to compute, so only the value of the function at certain points is known. This motivates the use of zeroth-order optimization (ZO) methods that do not utilize any derivatives of f. In certain applications such as online optimization and reinforcement learning, we may only be able to query f once per time step, which gives rise to single-point zeroth-order optimization (SZO) methods that only evaluate f once per iteration.

Using ideas from extremum seeking dynamics and control theory, the authors propose a SZO method called High/Low-pass Filter SZO (HLF-SZO), which they claim attains better performance on both convex and nonconvex problems compared to other SZO methods such as vanilla SZO [2] and residual SZO [3], while achieving similar performance to multi-point methods such as two-point ZO [4]. Due to the lack of publicly available code, we aim to reproduce experimental results from the paper using our own implementations of the methods to verify the claims by the authors, which are stated in detail in the scope of reproducibility below.

2 Scope of reproducibility

The main theoretical claim made in the paper is that HLF-SZO achieves a better iteration complexity compared to other SZO methods. More specifically, for minimizing a convex function with domain $\mathbb{R}^d$ to an optimality gap of ϵ, HLF-SZO admits a $\mathcal{O}(d^{3/2}/\epsilon^{3/2})$ iteration complexity, which is an improvement on the $\mathcal{O}(d^2/\epsilon^3)$ iteration complexity of vanilla SZO and the $\mathcal{O}(d^3/\epsilon^{3/2})$ iteration complexity of residual SZO, albeit worse than the $\mathcal{O}(d/\epsilon)$ complexity of two-point ZO.

In this report, we focus on reproducing both the theoretical results and numerical experiments that imply a "ranking" of the algorithms. In particular, we verify

(Claim 1) HLF-SZO has smaller variance and faster convergence compared to vanilla SZO for convex problems (logistic regression) (Figure 2 and Theorem 4.2 in original paper).

(Claim 2) HLF-SZO achieves faster convergence compared to residual SZO for convex problems (logistic regression, ridge regression) and nonconvex problems (Beale function) (Figure 4 and Theorem 4.2 in original paper).

(Claim 3) HLF-SZO achieves comparable (yet slightly slower) convergence to two-point ZO for convex problems (logistic regression, ridge regression) and nonconvex problems (Beale function) (Figure 4 in original paper).

Figures 3, 5 and Theorem 4.3 from the original paper are not reproduced in this report.

3 Methodology

Since the authors did not make their code publicly available, we implemented each algorithm as well as the figure generation code from their descriptions in the paper. We ran all code in a Google Colab notebook in Python, and since the problem sizes were relatively small, the code ran quickly on a laptop CPU.

3.1 Model descriptions

We describe all ZO algorithms that are discussed in the paper. In each of the methods, f is the function to be optimized.

Vanilla SZO. We repeatedly update $\mathrm{x_k} \in \mathbb{R}^d$ for $k \geq 1$ as

$$\mathrm{x_{k+1}} = \mathrm{x_k} - \eta \cdot \frac{d}{r} f(\mathrm{x_k} + r\mathrm{u_k})\mathrm{u_k} \tag{1}$$

where $\mathrm{u_k}$ are i.i.d. uniformly sampled from the $d-1$ dimensional unit sphere, η is the step size, and $r > 0$ is the smoothing radius.

Two-point ZO. We repeatedly update $\mathrm{x_k} \in \mathbb{R}^d$ for $k \geq 1$ as

$$\mathrm{x_{k+1}} = \mathrm{x_k} - \eta \frac{d}{2r}(f(\mathrm{x_k} + r\mathrm{u_k}) - f(\mathrm{x_k} - r\mathrm{u_k}))\mathrm{u_k}. \tag{2}$$

All parameters are identical to those from Vanilla SZO.

The paper's innovation involves interpreting vanilla SZO as a discrete extremum seeking (ES) control problem and integrating either a high-pass filter (giving HF-SZO), low-pass filter (giving LF-SZO), or both (giving HLF-SZO).

HF-SZO. We repeatedly update $\mathrm{x_k} \in \mathbb{R}^d$ for $k \geq 1$ as

$$\begin{aligned} z_k &= (1-\beta)z_{k-1} + f(\mathrm{x_k} + r\mathrm{u_k}) - f(\mathrm{x_{k-1}} + r\mathrm{u_{k-1}}) \\ \mathrm{x_{k+1}} &= \mathrm{x_k} - \eta \cdot \frac{d}{r} z_k \mathrm{u_k} \end{aligned} \tag{3}$$

where z_k is an intermediate scalar variable, $\beta \in (0, 2)$ is a parameter controlling the scaling of z_k; all other parameters are identical to those of the previous methods. We initialize $z_0 = 0$. We note that for implementation, we save the previous evaluation of f to query f only once per iteration.

Residual SZO. Residual SZO is equivalent to HF-SZO with $\beta = 1$.

LF-SZO. We repeatedly update $\mathrm{x_k} \in \mathbb{R}^d$ for $k \geq 1$ as

$$\mathrm{x_{k+1}} = \mathrm{x_k} - \eta \frac{d}{r} f(\mathrm{x_k} + r\mathrm{u_k})\mathrm{u_k} + \alpha(\mathrm{x_k} - \mathrm{x_{k-1}}) \tag{4}$$

where $\alpha \in [0, 1)$ is a momentum term; all other parameters are identical to those of the previous methods..

HLF-SZO. We repeatedly update $\mathrm{x_k} \in \mathbb{R}^d$ for $k \geq 1$ as

$$\begin{aligned} z_k &= (1-\beta)z_{k-1} + f(\mathrm{x_k} + r\mathrm{u_k}) - f(\mathrm{x_{k-1}} + r\mathrm{u_{k-1}}) \\ \mathrm{x_{k+1}} &= \mathrm{x_k} - \eta \cdot \frac{d}{r} z_k \mathrm{u_k} + \alpha(\mathrm{x_k} - \mathrm{x_{k-1}}). \end{aligned} \tag{5}$$

which is precisely the combination of HF-SZO and LF-SZO.

3.2 Datasets

All experiments reproduced from the original paper utilize simulated data to formulate each optimization problem - here we outline they are generated.

Logistic Regression (Figures 2 & 4, Extension 2). We generate N many observations $\{\mathbf{A}_i\}_{i=1}^N \subset \mathbb{R}^d$ where the entries of $\mathbf{A}_i$ are drawn i.i.d. from $\mathcal{U}(-1,1)$. Labels $\{y_i\}_{i=1}^N$ are generated as $y_i = \text{sign}(A_i^\top \mathbf{x}_* + \epsilon_i)$ where $\mathbf{x}_* = 0.5 \cdot \mathbf{1}_d$ is the ground truth and $\epsilon_i \sim \mathcal{U}(-0.5, 0.5)$ is some label noise. For Figure 2 and extension 2 we use $d = 2$ and $N = 200$, whereas for Figure 4 we use $d = 50$ and $N = 1000$.

Ridge Regression (Figure 4). We sample entries of a matrix $\mathbf{H} \in \mathbb{R}^{1000\times 50}$ i.i.d. from $\mathcal{N}(0,1)$, and $\mathbf{b} = \mathbf{H}\mathbf{x}_* + \epsilon$ where $\mathbf{x}_* = 0.5 \cdot \mathbf{1}_{50}$ and $\epsilon \sim \mathcal{N}(\mathbf{0}, 0.1\,\mathbf{I}_{50})$.

Circles (Extension 1). We construct a simple non-linear dataset (see Figure 3a) with 2 classes and $N = 500$ samples in total (250 samples for each class). Each sample $a_i \in \mathbb{R}^2$ represents a point on the circle of radius 3 or 5 for class -1 and 1, respectively.

MNIST (Extension 1). We sample 400 examples of the 0 and 1 digits respectively from the MNIST [5] dataset (see Figure 3b), which is sourced from the PyTorch [6] module `torchvision.datasets`.

3.3 Hyperparameters

The hyperparameters for all methods include the step size η, and the smoothing radius r for all zeroth order methods. For the high-pass/lower-pass filter methods, we also have the momentum term α and the residual term β. The authors of the paper claim they "manually optimize" the step size η to achieve the "fastest convergence" for each of the methods, which the authors clarified was done by gradually increasing the step size until the method diverged. The authors also stated that the radius r was manually chosen to be stable (i.e., no numerical or precision errors) but otherwise had little impact on the empirical performance. The momentum term α is set to 0.9 in all experiments and extensions, which the authors justify is a common setting in the literature. The residual term β is set to 1 in all experiments and extensions, which is theoretically optimal based on the convergence analysis presented in the paper. The radius r is set to be 0.1 for all reproductions and extensions (except for the Beale function problem, where $r = 0.01$). We report the step sizes chosen for each experiment and method in Table 1. For extension 1, we select the step size via grid search over 10 log-spaced values between 10^{-4} and 5 for each method, choosing the step size with fastest convergence. For extension 2, we use the same step size for HF-SZO as in the Figure 2 reproduction, whereas we manually tune the 2nd order HF-SZO method to achieve the fastest convergence (i.e., same procedure as the authors) for fair comparison.

	Vanilla	LF	HF	HLF	Residual	Two-Point	GD	2nd Order HF
Logistic Regression (Fig. 2)	5×10^{-4}	5×10^{-5}	0.3	0.05	–	–	–	–
Logistic Regression (Fig. 4a)	–	–	–	1.5×10^{-2}	4.5×10^{-2}	0.7	–	–
Ridge Regression (Fig. 4b)	–	–	–	10^{-6}	2.4×10^{-6}	2×10^{-5}	–	–
Beale Function (Fig. 4c)	–	–	–	10^{-6}	2.4×10^{-6}	2×10^{-5}	–	–
Circles (Ext. 1)	–	–	–	5×10^{-3}	5×10^{-2}	0.1	0.5	–
MNIST (Ext. 1)	–	–	–	10^{-4}	10^{-4}	0.01	1	–
Logistic Regression (Ext. 2)	–	–	0.3	–	–	–	–	0.6

Table 1. Step size η for each combination of experimental set-up and method.

3.4 Experimental setup and code

Reproducing Figure 2. We solve the logistic regression problem

$$\min_{x\in\mathbb{R}^d} f(x) := \frac{1}{N}\sum_{i=1}^{N} \log(1 + \exp(-y_i A_i^\top \mathbf{x}))$$

using the data described in Section 3.2. We apply the vanilla SZO, LF-SZO, HF-SZO, and HLF-SZO methods, initializing with $\mathbf{x}_0 = \mathbf{0}$. To compute the optimality gap $f(\mathbf{x}_\mathbf{k}) - f(\mathbf{x}^*)$ at each iteration, we calculate the optimal solution $\mathbf{x}^*$ of f via second-order BFGS using the Python package `scipy.optimize` - we note that the authors did not specify how $\mathbf{x}^*$ is computed in their results. For each method, 200 trials are performed and the mean of the optimality gap $f(\mathbf{x}_\mathbf{k}) - f(\mathbf{x}^*)$ is plotted along with a 80% confidence interval to observe variance. We run 10^5 iterations for vanilla SZO and LF-SZO since they are slower to converge, whereas we only run 500 iterations for HF-SZO and HLF-SZO.

Reproducing Figure 4. We apply residual SZO, HLF-SZO, and two-point ZO methods, initializing with $\mathbf{x}_0 = \mathbf{0}$. For each method, 200 trials are performed and the mean of the optimality gap $f(\mathbf{x}_\mathbf{k}) - f(\mathbf{x}^*)$ is plotted on a log scale along with a 80% confidence interval to observe variance. We run 5000 iterations of each method. For the first panel, we solve the same logistic regression problem as for Figure 2 with slightly different data as described in Section 3.2. For the second panel, we solve the ridge regression problem

$$\min_{x \in \mathbb{R}^d} f(x) := \frac{1}{2}\|\mathbf{b} - H\mathbf{x}\|_2^2 + \frac{c}{2}\|\mathbf{x}\|_2^2$$

using the data described in Section 3.2. For the third panel, we minimize the Beale function given by

$$\min_{x \in \mathbb{R}^2} f(x) := (1.5 - x_1 + x_1 x_2)^2 + (2.25 - x_1 + x_1 x_2^2)^2 + (2.625 - x_1 + x_1 x_2^3)^2$$

which is a nonconvex function with global minimum at $x^* = (3, 0.5)^\top$.

Extension 1. We train a 2-layer neural network using logistic loss, namely we solve

$$\min_{\theta} f(\theta) := \frac{1}{N}\sum_{i=1}^{N} \log(1 + \exp(-y_i h_\theta(a_i)))$$

where $h_\theta(a) = W_2\sigma(W_1 a + b_1) + b_2$ (σ is ReLU activation) and θ is a vectorized concatenation of all network weights $\{W_1, W_2, b_1, b_2\}$. We consider two different datasets: Circles and MNIST, which are described in Section 3.2. For the former, we have input dimension size 2 and hidden layer width 4, whereas for the latter we have input dimension size 784 and hidden layer width 10. We apply gradient descent, residual SZO, HLF-SZO, and two-point ZO methods, where network weights θ are initialized using the default initializations for `torch.nn.Linear` layers in PyTorch.

Extension 2. We solve the same logistic regression problem as for Figure 2 and apply HF-SZO and second order HF-SZO (derived in Appendix C), initializing with $\mathbf{x}_0 = \mathbf{0}$. For each method, 200 trials are performed and the mean of the optimality gap $f(\mathbf{x}_\mathbf{k}) - f(\mathbf{x}^*)$ is plotted on a log scale along with a 80% confidence interval to observe variance. We run 200 iterations of each method.

3.5 Computational requirements

All experiments were carried out using a Lenovo ThinkPad T14s Gen 3 with an AMD Ryzen 6750u (8 core) processor and 32GB DDR5-6000Mhz RAM. This is on Fedora 36 running Linux kernel 5.19. No GPUs were used for any experiments. The most time consuming aspect of reproducing the experiments is running each method 200 times, but generating all figures (including the extension) required less than an hour of computation time.

4 Results

Overall, we were able to reproduce both Figures 2 and 4 of the original paper with almost no visual discernible differences. Since these figures tend to support the 3 claims outlined in our scope of reproducibility, our findings also back these claims. On the other hand, our extension experiment implies that on slightly more “difficult” nonconvex problems, the faster convergence of HLF-SZO compared to residual SZO may not be as pronounced as suggested by the experiments in the original paper.

4.1 Results reproducing original paper

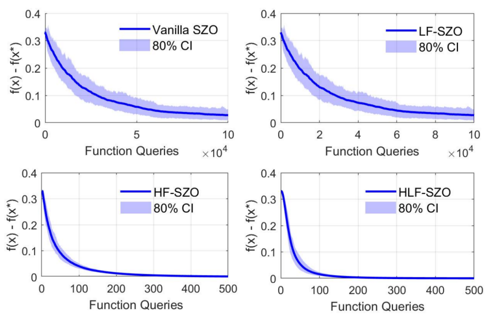
Result 1 (Reproducing Figure 2) — In Figure 1, we replicate Figure 2 from the paper, which compares the convergence speed and variance of different variations of the HLF-SZO algorithm, including strictly low-pass or high-pass variations, as well as the vanilla SZO method for solving a logistic regression problem. Our results are in agreement with the original figure shown in the paper. More specifically, we see that HLF-SZO takes orders of magnitude less function queries (iterations) to converge to the global minimizer compared to both Vanilla SZO, supporting claim 1. Moreover, observe that the variance of HLF-SZO (confidence interval) is a tighter region than the vanilla SZO, further supporting claim 1. This figure also functions somewhat as an ablation study, in the sense that integrating both high-pass and low-pass components gives faster convergence and lower variance than strictly using one or the other.

Result 2 (Reproducing Figure 4) — In Figure 2, we replicate Figure 4b from the paper, which compares the convergence speed and variance of HLF-SZO with established ZO methods such as residual SZO and two-point ZO for the convex ridge regression problem. In Appendix A, we also replicate Figures 4a and 4c with the same methods for both convex logistic regression and the nonconvex Beale function. Once again, our results are in agreement with the original figure shown in the paper. More specifically, we see that HLF-SZO converges faster in fewer function queries across all problems compared to residual SZO, supporting claim 2. Furthermore, HLF-SZO has comparable convergence speed to two-point ZO (the x-axis in these plots accounts for the fact that two-point ZO requires two function queries per iteration), supporting claim 3. In particular, for the ridge regression case, the initial slope of the HLF-SZO trace matches that of the two-point ZO trace.

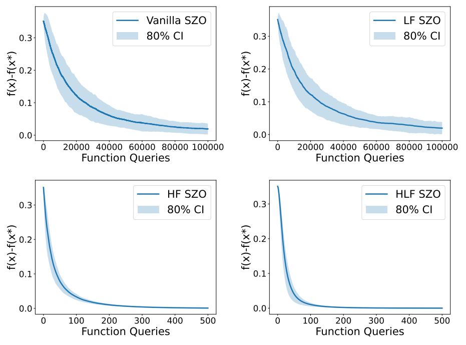
Result 3 (Reproducing Theorem 4.2) — In Appendix B, we fill in gaps of the proof of Theorem 4.2, which specifically shows that HLF-SZO requires $\mathcal{O}(d^{3/2}/\epsilon^{3/2})$ iterations to achieve an expected optimality gap of ϵ for convex problems of dimension d, which is an improvement on the $\mathcal{O}(d^2/\epsilon^3)$ iteration complexity of vanilla SZO and the $\mathcal{O}(d^3/\epsilon^{3/2})$ iteration complexity of residual SZO. This result supports both claims 1 and 2.

4.2 Results beyond original paper

Additional Result 1 (Zeroth Order Neural Network Training) — In Theorem 4.3, the authors give a convergence guarantee using the HLF-SZO method for nonconvex problems, yet the numerical results focus mainly on convex problems. They provide some experiments on the nonconvex Beale function as described in Section 3.4 of our report, but this is a relatively synthetic and low-dimensional problem, so it is difficult to assess the method for more realistic nonconvex problems of interest. To this end, we apply the various zeroth order optimization methods discussed here to the highly difficult nonconvex problem of optimizing a two-layer neural network for two different datasets. We report all technical details in Section 3.4. We plot the training loss vs. number of forward passes in Figure 3 for the Circles and MNIST datasets respectively. We observe in both figures that the convergence speed of gradient descent is much faster compared with all other zeroth order methods. On the other hand, the performance of HLF-SZO is only slightly worse than two-point SZO, which supports claim 3. Interestingly, while HLF-SZO performs similarly to residual SZO on the Circles dataset, there is a large gap in performance for the MNIST dataset, which supports claim 2.

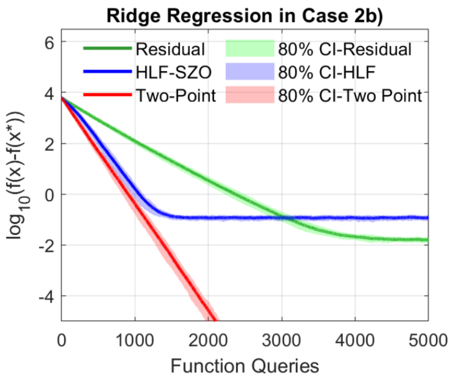


(a) Original result

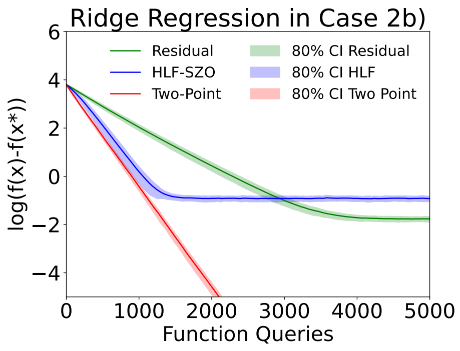


(b) Our result

Figure 1. The convergence results of various SZO methods on the logistic regression problem.

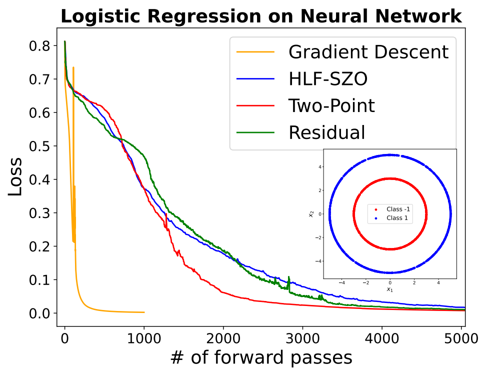


(a) Original result

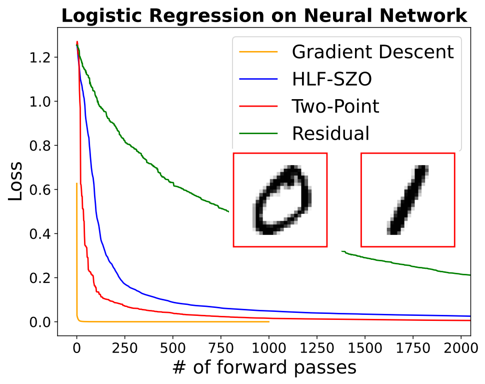


(b) Our result

Figure 2. The convergence results of various SZO methods on the ridge regression problem.



(a) Circles dataset (blue and red classes shown above).



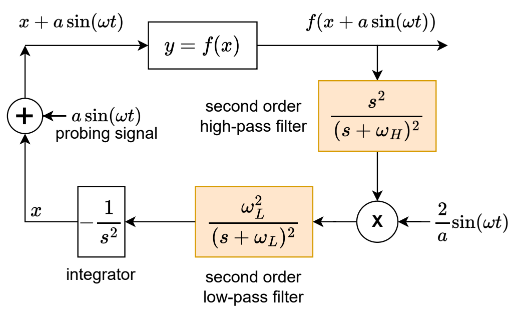
(b) MNIST dataset (digit classes shown above).

Figure 3. Convergence results of GD and various SZO methods for a two-layer neural network.

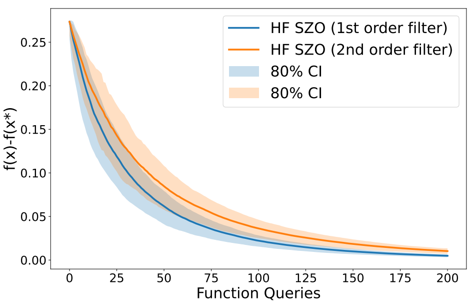
Additional Result 2 (Higher Order Filters) — The authors of the paper derive HLF-SZO by applying first order low and high-pass filters to the continuous-time dynamics of vanilla SZO. A natural question to ask is: *can we apply higher order filters to yield the same benefits?* Here we derive a new second order HLF-SZO method from second order high pass and low pass filters and compare to the original HLF-SZO method. The equation below is the result of our derivation shown in Appendix C.

$$\begin{aligned} z_k &= \frac{1}{1+2\beta}((2+2\beta)z_{k-1} - (1+\beta^2)z_{k-2} + f(x_k + ru_k) \\ &\quad + f(x_{k-2} + ru_{k-2}) - 2f(x_{k-1} + ru_{k-1})) \\ x_{k+1} &= x_k - \eta\frac{d}{r}z_k u_k \end{aligned}$$

In Figure 4a, we illustrate the block diagram for the second order HLF-SZO - we derive the exact method in Appendix C. In Figure 4b, we compare the second order HF-SZO to the original (first order) HF-SZO method on the same logistic regression problem from Figure 1. We only consider the high pass component since it yielded most of benefit in Figure 1. We see that the second order HF SZO performs very similarly (with almost identical variance) compared to the first order HF SZO, albeit with slightly slower convergence. This suggests that higher order filters can provide similar variance reduction effects as with the first order filter introduced in the paper.



(a) Block diagram of second order HLF-SZO.



(b) Convergence result on logistic regression problem.

Figure 4. Second order filter effects on HLF-SZO algorithm convergence.

5 Discussion

The main strength of our approach is that we were able to reproduce the figures to be nearly visually identical without any code provided by authors or instructions beyond what was reported in the paper. This indicates that the results are easily reproducible and that claims outlined in our scope of reproducibility are generally well supported.

For claim 1, even though the comparison to vanilla SZO is only made for a small logistic regression problem, the dramatic difference in convergence speed shown in Figure 1 for vanilla SZO (10^5 iterations) vs. HLF-SZO (400 iterations) is quite convincing. Along with the difference in theoretical iteration complexity, we believe that claim 1 is well-supported.

For claim 2, it seems that Figure 5 & 6, along with the theoretical iteration complexity gap, support the idea that HLF-SZO converges faster than residual SZO. Figure 3b also suggests that HLF-SZO can convergence much quicker than residual SZO for certain problems. However, for the same neural network trained on a different dataset, the performance gap may vanish, as seen in Figure 3a. Furthermore, in the ridge regression problem in Figure 2, even though HLF-SZO has a faster rate of convergence initially, residual SZO achieves a smaller optimality gap eventually. We believe that this behavior can be alleviated by using a diminishing step size for both methods, so that both methods can gradually approach arbitrarily small optimality gaps rather than flattening out. Overall, we believe that claim 2 is mostly supported but can depend heavily on the specific optimization problem.

Finally, we believe that claim 3, namely that HLF-SZO has comparable performance to two-point ZO, is not entirely clear. Again, as shown in Figure 2 for ridge regression, two-point ZO has an arbitrarily decreasing optimality gap whereas HLF-SZO flattens out. Moreover, on our neural network experiment, two-point ZO converges to zero training loss noticeably quicker (about 1000 iterations). Combined with the fact that two-point ZO has a strictly better theoretical iteration complexity compared to HLF-SZO, we cannot say for certain that claim 3 is well supported.

Overall, we believe that the paper is technically sound and the claims are mostly supported by various experiments, including our own additional simulations. Our extension experiment seems to suggest that filtering the time dynamics of iterative optimization algorithms to improve convergence may be a more general idea than simply the method proposed in the original paper, which is an exciting direction for future work.

5.1 What was easy

Since the problem sizes are relatively small (dimensions of vectors and matrices were small) and the algorithms were fairly simple (only several lines long each), we were able to reimplement them from scratch without much problems and run the code quickly.

5.2 What was difficult

One part of the reproduction process that was difficult involved recreating the graphs in figures 2 and 4. This was because in the paper, the vertical scale showed some variant of $f(x) - f(x^*)$. The paper in all cases defined x^* as some fixed point earlier, but in reality, when creating the plots, x^* was intended to be the optimal point of the optimization point which was different from the paper's specified point. This initially led to confusion with algorithms that seemingly did not converge.

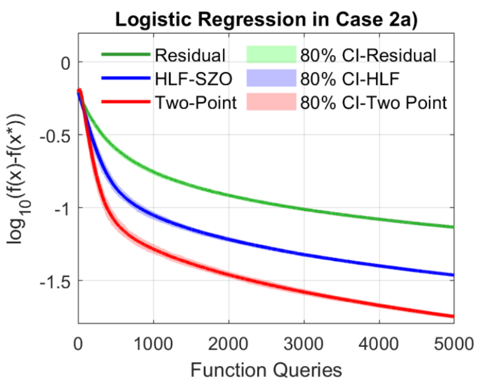
5.3 Communication with original authors

We have contacted the authors about details regarding the "manual search" procedure for finding the various step sizes (as well as the smoothing radius parameter) that were reported in the original paper. They explained that the step sizes were found by incrementally increasing the step size until the method diverged, thus giving each method the best possible performance. They also claimed that the smoothing radius was selected so that the method did not have any numerical errors but otherwise did not have much effect on the convergence results.

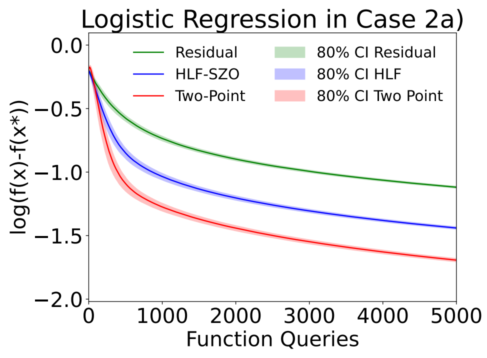
References

1. X. Chen, Y. Tang, and N. Li. "Improve Single-Point Zeroth-Order Optimization Using High-Pass and Low-Pass Filters." In: **Proceedings of the 39th International Conference on Machine Learning**. Ed. by K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato. Vol. 162. Proceedings of Machine Learning Research. PMLR, July 2022, pp. 3603–3620.
2. A. V. Gasnikov, E. A. Krymova, A. A. Lagunovskaya, I. N. Usmanova, and F. A. Fedorenko. "Stochastic online optimization. Single-point and multi-point non-linear multi-armed bandits. Convex and strongly-convex case." In: **Automation and Remote Control** 78 (2017), pp. 224–234.
3. Y. Zhang, Y. Zhou, K. Ji, and M. M. Zavlanos. **A New One-Point Residual-Feedback Oracle For Black-Box Learning and Control**. 2020.
4. Y. Nesterov and V. G. Spokoiny. "Random Gradient-Free Minimization of Convex Functions." In: **Foundations of Computational Mathematics** 17 (2017), pp. 527–566.
5. Y. LeCun, C. Cortes, and C. Burges. **Mnist handwritten digit database. AT&T Labs**. 2010.
6. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. "Pytorch: An imperative style, high-performance deep learning library." In: **Advances in neural information processing systems** 32 (2019).

A Additional Reproductions

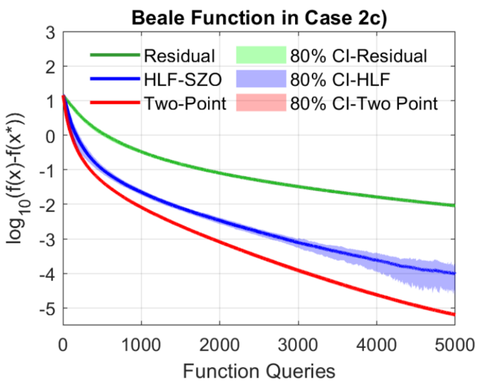


(a) Original result

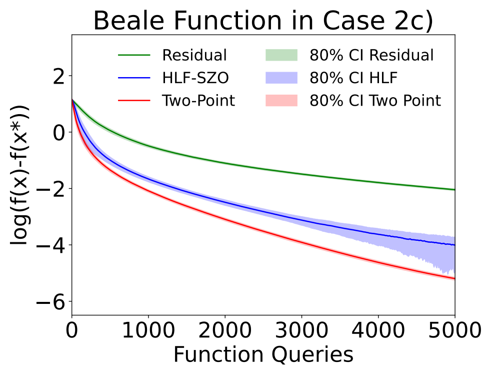


(b) Our result

Figure 5. The convergence results of the various SZO methods on the logistic regression problem.



(a) Original result



(b) Our result

Figure 6. The convergence results of various SZO methods on the Beale function.

B Convergence Proof

See the following table for math notation.

α, β	high-pass filter parameter, low-pass filter parameter
$\tilde{\beta} = 1 - \|1-\beta\|$	simplified notation of β
$\theta = 1 - \frac{4\eta^2 d^2 G^2 (1+\alpha^2)}{\tilde{\beta}^2 r^2 (1-\alpha^2)^2}$	temporary parameter for notation simplicity
$f(x), \nabla f(x)$	objective function, gradient of objective function
d, T	dimension space for optimization variable, number of iterations
$x^*, f(x^*)$	optimal value, optimal objective function value
η	descent step size from current iterate to future iterate
L, G	Lipschitz constant of $\nabla f(x)$, Lipschitz constant of $f(x)$
$r, f_r(x)$	smoothing radius, smoothed version of $f(x)$
g_k, z_k	current search direction with momentum, past function evaluations
p_k, w_k	momentum term, updated iteration value using past momentum terms

In the convex setting, let the low-pass and high-pass parameters fall in range $\alpha \in [0, 1), \beta \in (0, 2)$. For a fixed number of iterations T and minimizer $\mathbf{x}^* \in \mathbb{R}^d$, choose the step size such that the following holds

$$\eta \leq \frac{(1-\alpha)(1-|1-\beta|)^2}{20LdT^{1/3}}$$

where d is the dimension of the input variable and L is the modulus of continuity of the gradient of the objective function (i.e. the Lipschitz constant of $\nabla f(\mathbf{x})$). Given this step size selected, you must pick a searching radius bounded by the following

$$\frac{4\eta dG}{(1-|1-\beta|)(1-\alpha)} \leq r \leq \frac{G}{LT^{1/3}}$$

where G is the modulus of continuity of the objective function (i.e. the Lipschitz constant of $f(\mathbf{x})$). Then, on average, $\bar{x_T} = \frac{1}{T}\sum_{k=1}^{T} x_k$ achieves the following convergence

$$\mathbb{E}[f(\bar{x_T})] - f(x^*) \leq \frac{3(1-\alpha)\|x_1 - x^*\|^2}{4\eta T} + \frac{3G^2}{2LT^{2/3}} + \mathcal{O}(\frac{d}{T})$$

From above, we can say that on average, $f(x_T)$ will converge to optimal $f(x^*)$ as $T \to \infty$ as seen on the right hand side that goes towards zero. To show this convergence, we will introduce the following notation that makes the problem more notation heavy but easier to deal with the proof.

$$g_k = \frac{d}{r} z_k u_k, \quad z_k = (1-\beta) z_{k-1} + f(x_k + ru_k) - f(x_{k-1} + ru_{k-1})$$

$$p_k = \alpha p_{k-1} + \eta g_k, \quad x_{k+1} = x_k - p_k$$

Notice that this notation above simplifies to the notation below used throughout the paper

$$z_k = (1-\beta)z_{k-1} + f(\mathrm{x_k} + r\mathrm{u_k}) - f(\mathrm{x_{k-1}} + r\mathrm{u_{k-1}})$$
$$\mathrm{x_{k+1}} = \mathrm{x_k} - \eta \cdot \frac{d}{r} z_k \mathrm{u_k} + \alpha(\mathrm{x_k} - \mathrm{x_{k-1}}).$$

Define vector $w_k = x_k - \alpha p_{k-1}/(1-\alpha)$ that contains current iterate and past iterate momentum terms such that the following can be expressed

$$w_{k+1} = x_{k+1} - \frac{\alpha}{1-\alpha} p_k = x_k - \frac{1}{1-\alpha} p_k$$

by plugging in definitions of w_k, x_k and rearranging terms. Moreover, this can further be manipulated to express

$$w_{k+1} = x_k - \frac{\alpha}{1-\alpha} p_{k-1} - \frac{\eta}{1-\alpha} g_k = w_k - \frac{\eta}{1-\alpha} g_k$$

Here the definitions are reused to express the future weight vector as a combination of the current weight vector and the scaled search direction vector. This is done by substituting x_{k+1} formula, using w_k definition, and p_k definition. Using this last formula, analyzing the L2 norm of this momentum vector and the optimal solution can give

$$\|w_{k+1} - x^*\|^2 = \|w_k - x^*\|^2 - \frac{2\eta}{1-\alpha}\langle w_k - x^*, g_k\rangle + \frac{\eta^2}{(1-\alpha)^2}\|g_k\|^2$$

using expansion of L2 norm such that it is expressed in terms of current w_k. What will be shown is that w_k converges on average to the optimal solution x^* . Remember that w_k is essentially the current iterate with some momentum terms. Notice that $g_k = \nabla f_r(x_k)$ since we are taking a difference in the objective function. Here $f_r(\cdot)$ means the average function value for all search directions in the given radius r. This is a smoothed version of $f(\cdot)$. Given this, on average, the L2 distance satisfies the following

$$\mathbb{E}[\|w_{k+1} - x^*\|^2] = \|w_k - x^*\|^2 - \frac{2\eta}{1-\alpha}\langle w_k - x^*, \nabla f_r(x_k)\rangle + \frac{\eta^2}{(1-\alpha)^2}\mathbb{E}[\|g_k\|^2]$$

Using the definition of the weight vector and expressions for the iterates one can show that $\langle w_k - x^*, \nabla f_r(x_k)\rangle = \langle x_k - x_{k-1}, \nabla f_r(x_k)\rangle - \frac{\alpha}{(1-\alpha)^2}\langle x_k - x_{k-1}, \nabla f_r(x_k)\rangle$. We can use this in the expression above. First, however, using the fact that the function is convex, by definition of convexity $-\langle x_k - x_{k-1}, \nabla f_r(x_k)\rangle \le -(f_r(x_k) - f_r(x_{k-1}))$. The same holds for the optimal solution i.e. $-\langle x_k - x^*, \nabla f_r(x_k)\rangle \le -(f_r(x_k) - f_r(x^*))$. Putting these three inequalities together in the expectation of the L2 norm, we get the final expression

$$\begin{aligned}\mathbb{E}[\|w_{k+1} - x^*\|^2] \le \mathbb{E}[\|w_k - x^*\|^2] - \frac{2\eta}{1-\alpha}\mathbb{E}[f_r(x_k) - f_r(x^*)] \\ - \frac{2\eta\alpha}{(1-\alpha)^2}\mathbb{E}[f_r(x_k) - f_r(x_{k-1})] + \frac{\eta^2}{(1-\alpha)^2}\mathbb{E}[\|g_k\|^2]\end{aligned}$$

The next part of the proof that follows uses Lemma A.3 that generally states that

$$\sum_{k=1}^{T}\mathbb{E}[\|g_k\|^2] \le \frac{5d}{\theta\tilde{\beta}^2}\sum_{k=1}^{T}\mathbb{E}[\|\nabla f(x_k)\|^2] + \frac{10Tr^2L^2d^2}{\theta\tilde{\beta}^2} + \frac{5d}{2\theta\tilde{\beta}}\|\nabla f(x_1)\|^2$$

for some constants $\beta = 1 - |1-\beta|, \theta = 1 - \frac{4\eta^2 d^2 G^2(1+\alpha^2)}{\tilde{\beta}^2 r^2 (1-\alpha^2)^2}$ that depend on step size, dimensionality, iterations, Lipschitz constant, and filter parameters. This means that on average, the past momentum terms are upper bounded by the gradient of the objective function. This will be useful to get the right had side of the $\mathbb{E}[\|w_{k+1} - x^*\|^2]$ above in terms of objective function. From a calculus class, recall that a telescoping sum is of the form $\sum_{k=1}^{n-1}(x_k - x_{k+1}) = x_1 - x_2 + x_2 - x_3 + x_3... = x_1 - x_n$. Notice that the series $w_1, ..., w_T$ for $k = 1, ..., T$ is a telescoping sum so we can focus on the first and last terms of w_k. Now the new expression becomes the following

$$\begin{aligned}\mathbb{E}[\|w_{T+1} - x^*\|^2] \leq \|w_1 - x^*\|^2 - \frac{2\eta}{1-\alpha}\sum_{k=1}^{T}\mathbb{E}[f_r(x_k) - f_r(x^*)] - \frac{2\eta\alpha}{(1-\alpha)^2}\mathbb{E}[f_r(x_T) - f_r(x_1)]\\ + \frac{10\eta^2 Ld}{(1-\alpha)^2\theta\tilde{\beta}^2}\sum_{k=1}^{T}\mathbb{E}[f(x_k) - f(x^*)] + \frac{5\eta^2 Ld}{(1-\alpha)^2\theta\tilde{\beta}}(f(x_1) - f(x^*))\\ + (\frac{2\eta T r^2 L}{1-\alpha} + \frac{2\eta\alpha r^2 L}{(1-\alpha)^2} + \frac{10T\eta^2 r^2 L^2 d^2}{(1-\alpha)^2\theta\tilde{\beta}^2})\end{aligned}$$

since $w_2, ..., w_T$ terms will cancel each other out. Using Lemma A.1, it was shown that $f_r(x) \geq f(x)$ for a convex function. Meaning that the smoothed version of the objective function upper bounds the original objective function. Given this, we can bound the previous equation so that everything is in terms of $f(\cdot)$

$$\begin{aligned}\mathbb{E}[\|w_{T+1} - x^*\|^2] \leq \|w_1 - x^*\|^2 - \frac{2\eta}{1-\alpha}\sum_{k=1}^{T}\mathbb{E}[f(x_k) - f(x^*)] - \frac{2\eta\alpha}{(1-\alpha)^2}\mathbb{E}[f(x_T) - f(x_1)]\\ + \frac{10\eta^2 Ld}{(1-\alpha)^2\theta\tilde{\beta}^2}\sum_{k=1}^{T}\mathbb{E}[f(x_k) - f(x^*)] + \frac{5\eta^2 Ld}{(1-\alpha)^2\theta\tilde{\beta}}(f(x_1) - f(x^*))\\ + (\frac{2\eta T r^2 L}{1-\alpha} + \frac{2\eta\alpha r^2 L}{(1-\alpha)^2} + \frac{10T\eta^2 r^2 L^2 d^2}{(1-\alpha)^2\theta\tilde{\beta}^2})\end{aligned}$$

Using the previous inequality $\mathbb{E}[\|w_{T+1} - x^*\|^2] \leq \|w_1 - x^*\|^2 + ...$ and the fact that $w_1 = x_1$ (no momentum first iteration) and $\mathbb{E}[\|w_{T+1} - x^*\|^2] \geq 0$ everything with expectations is moved to the left hand side to express

$$\begin{aligned}(1 - \frac{5\eta Ld}{(1-\alpha)\theta\tilde{\beta}^2})\sum_{k=1}^{T}\mathbb{E}[f(x_k) - f(x^*)] + \frac{2\eta\alpha}{(1-\alpha)^2}\mathbb{E}[f(x_T) - f(x_1)]\\ \leq \frac{5\eta^2 Ld}{(1-\alpha)^2\theta\tilde{\beta}}(f(x_1) - f(x^*)) + (\frac{2\eta T r^2 L}{1-\alpha} + \frac{2\eta\alpha r^2 L}{(1-\alpha)^2} + \frac{10T\eta^2 r^2 L^2 d^2}{(1-\alpha)^2\theta\tilde{\beta}^2})\end{aligned}$$

given that $\mathbb{E}[f(x_T) - f(x^*)] \leq \mathbb{E}[f(x_T) - f(x_1)]$ since an iterate at T will be closer than the first initialization. This equation above will be used in the following inequality. Using inequality properties, it is known that adding any positive number to another number makes it larger so it follows that

$$\begin{aligned}(1 - \frac{5\eta Ld}{(1-\alpha)\theta\tilde{\beta}^2})\frac{1}{T}\sum_{k=1}^{T}\mathbb{E}[f(x_k) - f(x^*)]\\ \leq (1 - \frac{5\eta Ld}{(1-\alpha)\theta\tilde{\beta}^2})\frac{1}{T}\sum_{k=1}^{T}\mathbb{E}[f(x_k) - f(x^*)] + \frac{\alpha}{1-\alpha}\frac{\mathbb{E}[f(x_T) - f(x^*)]}{T}\end{aligned}$$

Putting these last two inequalities together results in the following conclusion that $\mathbb{E}[f(x_k) - f(x^*)]$ is bounded by how good the initialization is, the filter parameters selected, the Lipschitz constants, the smoothing radius selected, and the number of iterations.

$$\begin{aligned}&(1-\frac{5\eta Ld}{(1-\alpha)\theta\tilde{\beta}^2})\frac{1}{T}\sum_{k=1}^{T}\mathbb{E}[f(x_k)-f(x^*)]\\&\le\frac{(1-\alpha)\|x_1-x^*\|^2}{2\eta T}+\frac{5\eta r^2L^2d^2}{(1-\alpha)\theta\tilde{\beta}^2}+r^2L+\frac{\alpha r^2L}{(1-\alpha)T}\\&+(\frac{\alpha}{1-\alpha}+\frac{5\eta Ld}{2(1-\alpha)\theta\tilde{\beta}})\frac{f(x_1)-f(x^*)}{T}\end{aligned}$$

Since η, r are selected based on the inequalities described in the beginning of this section, then the temporary parameters defined $(\theta, \tilde{\beta})$ are bounded by constants such that $\theta \geq 1-\frac{1}{4}\frac{(1+\alpha^2)}{(1+\alpha)^2} \geq \frac{3}{4}$ and $1-\frac{5\eta Ld}{(1-\alpha)\theta\tilde{\beta}^2} \geq 1-\frac{1}{3T^{1/3}} \geq \frac{2}{3}$. Given these bounds, the previous equation simplifies to reach the final conclusion that

$$\frac{1}{T}\sum_{k=1}^{T}\mathbb{E}[f(x_k)-f(x^*)] \leq \frac{3(1-\alpha)\|x_1-x^*\|^2}{4\eta T}+\frac{3G^2}{2LT^{2/3}}+\mathcal{O}(\frac{d}{T})$$

As seen above, on average, the sequence generated by HLF-SZO converges to x^* since the right hand side has a decay factor $\frac{1}{T}$ so as $T \to \infty$, then the right hand side goes towards zero.

Lemma A.1 — Denote $\mathbb{B} := \{x \in \mathbb{R}^d : \|x\| \leq 1\}$. For all $k \geq 1$, we have

$$\mathbb{E}[g_k|\mathcal{F}_k] = \nabla f_r(x_k),$$

where $f_r : \mathbb{R}^d \to \mathbb{R}$ is defined by $f_r(x) := \mathbb{E}_{y\sim\mathrm{Unif}(\mathbb{B}_d)}[f(x+ry)]$. Moreover, for all $x \in \mathbb{R}^d$,

$$|f_r(x)-f(x)| \leq \frac{1}{2}Lr^2, \qquad \|\nabla f_r(x)-\nabla f(x)\| \leq Lr,$$

and if f is convex, then $f_r(x) \geq f(x)$.

Lemma A.2 — We have

$$\sum_{k=1}^{T}\mathbb{E}[|z_k|^2] \leq \frac{5r^2}{\tilde{\beta}^2 d}\sum_{k=1}^{T-1}\mathbb{E}[\|\nabla f(x_k)\|^2]+\frac{2G^2}{\tilde{\beta}^2}\sum_{k=1}^{T-1}\mathbb{E}[\|p_k\|^2]+\frac{10Tr^4L^2}{\tilde{\beta}}+\frac{5r^2}{2\tilde{\beta}d}\|\nabla f(x_1)\|^2.$$

Lemma A.3 — Suppose the quantity

$$\theta := 1-\frac{4\eta^2d^2G^2(1+\alpha^2)}{\tilde{\beta}^2r^2(1-\alpha^2)^2}$$

is positive. Then we have

$$\begin{aligned}&\sum_{k=1}^{T}\mathbb{E}[\|g_k\|^2] \leq \frac{5d}{\theta\tilde{\beta}^2}\sum_{k=1}^{T}\mathbb{E}[\|\nabla f(x_k)\|^2]+\frac{10Tr^2L^2d^2}{\theta\tilde{\beta}^2}+\frac{5d}{2\theta\tilde{\beta}}\|\nabla f(x_1)\|^2,\\&\sum_{k=1}^{T}\mathbb{E}[\|p_k\|^2] \leq \frac{2\eta^2(1+\alpha^2)}{(1-\alpha^2)^2}\sum_{k=1}^{T}\mathbb{E}[\|g_k\|^2].\end{aligned}$$

C Second-Order Filter Derivation

We follow a similar derivation strategy as in the original paper. The second order HLF-SZO method is derived by squaring first order filter transfer functions. For the highpass filter, we have

$$L(z) = \frac{s^2}{(s+\omega_H)^2}L(f),$$

leading to $L(z)(s^2+2\omega_H s+\omega_H^2) = s^2 L(f)$ so that $\ddot{z}+2\omega_H\dot{z}+\omega_H^2 z = \ddot{f}$. We can approximate second derivatives as $\ddot{z} = \frac{z_k+z_{k-2}-2z_{k-1}}{\delta^2}$ so that this becomes

$$\frac{z_k+z_{k-2}-2z_{k-1}}{\delta^2}+2\omega_H\frac{z_k-z_{k-1}}{\delta}+\omega_H^2 z_{k-2} = \frac{f_k+f_{k-2}-2f_{k-1}}{\delta^2},$$

which after simplifying becomes

$$z_k = \frac{1}{1+2\omega_H\delta}((2+2\omega_H\delta)z_{k-1}-(1+\delta^2\omega_H^2)z_{k-2}+f_k+f_{k-2}-2f_{k-1})).$$

After making the substitution $\beta = \delta\omega_H$, and fully writing the expressions f_k, we have the second order highpass filter update rule given by

$$z_k = \frac{1}{1+2\beta}((2+2\beta)z_{k-1}-(1+\beta^2)z_{k-2}+f(x_k+ru_k)+f(x_{k-2}+ru_{k-2})-2f(x_{k-1}+ru_{k-1})),$$

and $x_{k+1} = x_k - \eta\frac{d}{r}z_k u_k$ as in the first order update rule.

For the second order lowpass filter, we similarly square the transfer function to get

$$L(z) = \frac{\omega_L^2}{(s+\omega_L)^2}L(f),$$

which after applying the same discrete time second derivative approximations simplifies to

$$\frac{y_k+y_{k+2}-2y_{k+1}}{\delta^2}+2\omega_L\frac{y_{k+1}-y_k}{\delta}+\omega_L^2 y_k = \omega_L^2 g.$$

Rearranging gives

$$y_{k+2} = (2-2\delta\omega_L)y_{k+1}+(2\delta\omega_L-\omega_L^2\delta^2-1)y_k+\delta^2\omega_L^2 g.$$

We also need to perform one other modification to the figure in the paper, which is to replace the single integrator with a double integrator whose transfer function is $\frac{1}{s^2}$. After doing this, we get $\ddot{x} = -y$. Therefore, we may apply the discrete time second derivative approximation formula to replace all the y terms in the above equation, and also let $\alpha = 1-\delta\omega_L$ and $\eta = \delta^2\omega_L$ to get

$$2x_{k+1}-x_k-x_{k+2} = 2\alpha(2x_k-x_{k-1}-x_{k+1})-\alpha^2(2x_{k-1}-x_{k-2}-x_k)+\eta^2 g,$$

which after rearrangement becomes our update rule given by

$$x_{k+2} = (2+2\alpha)x_{k+1}+(-1-4\alpha-\alpha^2)x_k+(2\alpha+2\alpha^2)x_{k-1}-\alpha^2 x_{k-2}-\eta^2 g.$$

Note that this can also be rewritten as

$$x_{k+2} = x_{k+1}+(1+2\alpha)(x_{k+1}-x_k)+(-2\alpha-\alpha^2)(x_k-x_{k-1})+\alpha^2(x_{k-1}-x_{k-2})-\eta^2 g,$$

which bares a close resemblance to the first order lowpass filter update rule. During the actual implementation, we may decrement each of the indices by one so that x_{k+1} is being updated.