Advances in Applied Mathematics 应用数学进展, 2026, 15(4), 450-463
Published Online April 2026 in Hans. https://www.hanspub.org/journal/aam
https://doi.org/10.12677/aam.2026.154173

Hans汉斯

基于滑动窗口的自适应学习率优化方法

靳唯一[1]，陆　莎[2*]

[1]南宁师范大学数学与统计学院，广西　南宁
[2]南宁师范大学广西应用数学中心，广西　南宁

收稿日期：2026年3月8日；录用日期：2026年4月2日；发布日期：2026年4月10日

摘　要

针对深度神经网络训练中梯度噪声干扰导致参数更新偏离和高曲率区域参数振荡的问题，给出一种在随机优化中能够快速响应局部梯度噪声变化的自适应学习率优化方法(SW-Adam)。该方法设计了滑动窗口方差估计和方差感知学习率调整两个协同机制：一方面，采用Welford在线算法对窗口内梯度赋予均匀权重$1/k$，将方差估计的响应延迟从传统指数加权移动平均(EMA)的$O\left(1/(1-\beta)\right)$ (约1000步)降至$O(k)$ (k为窗口长度，通常取50)；另一方面，通过指数衰减函数建立学习率与方差估计的负相关关系，在高噪声区域自动降低学习率以抑制振荡，在低噪声区域提高学习率以加速收敛。在60维带噪声旋转Rastrigin函数上的初步数值实验结果表明，SW-Adam的最终函数值为130.80，相比AdamW降低77.6%，相比Shampoo降低88.1%；方差估计波动系数相比Adam降低8.6%；虽然SW-Adam的单步计算耗时略高于Adam，但其达到相同目标函数值所需的总迭代次数更少；Wilcoxon秩和检验证实与各基线算法的性能差异具有统计显著性($p < 0.001$)。消融实验表明三个组件各具独立贡献，其中滑动窗口方差估计贡献最大(移除后性能下降202.3%)，方差感知指数衰减和学习率裁剪分别提供关键的信号转化和边界稳定保障。在CIFAR-10 + ResNet-18深度学习任务上，SW-Adam取得93.47%的测试准确率，泛化差距为3.21个百分点，验证了该方法在深度学习任务上的泛化能力。滑动窗口方差估计机制能够较好地反映局部梯度噪声水平，与方差感知学习率调整策略协同工作，在保持计算高效的同时有效改善收敛性能。

关键词

深度学习，自适应学习率，滑动窗口，方差估计，梯度优化

Sliding-Window Based Adaptive Learning Rate Optimization Method

Weiyi Jin[1], Sha Lu[2*]

[1]School of Mathematics and Statistics, Nanning Normal University, Nanning Guangxi
[2]The Center for Applied Mathematics of Guangxi, Nanning Normal University, Nanning Guangxi

[*]通讯作者。

文章引用: 靳唯一, 陆莎. 基于滑动窗口的自适应学习率优化方法[J]. 应用数学进展, 2026, 15(4): 450-463.
DOI: 10.12677/aam.2026.154173

Received: March 8, 2026; accepted: April 2, 2026; published: April 10, 2026

Abstract

To address the problems of parameter update deviation caused by gradient noise and parameter oscillation in high-curvature regions during deep neural network training, this study proposes an adaptive learning-rate optimization method, termed SW-Adam, that can rapidly respond to local variations in gradient noise under stochastic optimization. The method incorporates two coordinated mechanisms: sliding-window variance estimation and variance-aware learning rate adjustment. On the one hand, the Welford online algorithm is employed to assign a uniform weight of $1/k$ to gradients within the window, thereby reducing the response delay of variance estimation from $O\left(1/\left(1-\beta\right)\right)$ in the conventional exponential moving average (EMA) approach (approximately 1000 steps) to $O(k)$, where k denotes the window length and is typically set to 50. On the other hand, an exponential decay function is used to establish a negative correlation between the learning rate and the variance estimate, so that the learning rate is automatically decreased in high-noise regions to suppress oscillations and increased in low-noise regions to accelerate convergence. Preliminary numerical experiments on the 60-dimensional noisy rotated Rastrigin function show that SW-Adam achieves a final function value of 130.80, representing a 77.6% reduction compared to AdamW and an 88.1% reduction compared to Shampoo; the variance estimation coefficient of variation decreases by 8.6% compared to Adam; although the single-step computation time of SW-Adam is slightly higher than that of Adam, it requires fewer total iterations to achieve the same objective function value; Wilcoxon rank-sum test confirms that the performance differences with all baseline algorithms are statistically significant ($p < 0.001$). Ablation experiments confirm the independent contribution of each component: the sliding-window variance estimation is the primary contributor (performance degrades by 202.3% upon removal), while the variance-aware exponential decay and learning rate clipping provide essential signal transformation and boundary stabilization, respectively. On the CIFAR-10 image classification task with ResNet-18, SW-Adam achieves a test accuracy of 93.47% with a generalization gap of 3.21 percentage points, demonstrating its generalization capability on practical deep learning tasks. The sliding-window variance estimation mechanism can effectively reflect local gradient noise levels, working synergistically with the variance-aware learning rate adjustment strategy to improve convergence performance while maintaining computational efficiency.

Keywords

Deep Learning, Adaptive Learning Rate, Sliding Window, Variance Estimation, Gradient Optimization

1. 引言

深度神经网络的训练本质上是高维非凸优化问题。随着模型规模从百万参数增长到数十亿参数[1]，优化器的设计对训练效率和最终性能的影响日益显著。自适应学习率优化算法通过梯度统计信息动态调整学习率，在深度学习领域得到广泛应用[2]。

自适应学习率算法的发展主要沿着两条技术路线推进。第一条是标量自适应路线：AdaGrad (Adaptive Subgradient Methods) [2]累积历史梯度平方 $G_t = \sum_{i=1}^{t} g_i^2$ 作为学习率分母，其对稀疏特征有效但学习率单

调递减；RMSProp (Root Mean Square Propagation) [3]引入指数加权移动平均 EMA (Exponential Moving Average)机制 $v_t = \beta v_{t-1} + (1-\beta) g_t^2$ 替代梯度累积求和，缓解了学习率持续衰减的问题；而 Adam (Adaptive moment estimation) [4]结合动量与 EMA 方差估计进行迭代，其迭代点 θ_t 的更新规则定义如下：

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, \quad v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2, \quad \theta_t = \theta_{t-1} - \eta \frac{m_t}{\sqrt{v_t} + \epsilon} \tag{1}$$

其中 m_t 为迭代过程中的梯度一阶矩估计，v_t 为梯度二阶矩估计，β_1、β_2 为衰减系数(典型取值分别为 0.9 和 0.999)，η 为全局步长，ϵ 为数值稳定项。Adam 因其稳健性和易用性成为当前最主流的优化器。

第二条是矩阵自适应路线：Anil [5]等(2021)提出可扩展二阶优化方法 Shampoo，通过矩阵预条件刻画参数间二阶相关性，提升了深度学习训练效率。Chen [6]等(2023)通过符号搜索发现 Lion，以更简洁的符号更新规则实现较强性能。Shazeer 和 Stern (2018) [7]提出 Adafactor，利用因子分解显著降低二阶统计量存储开销，增强了大模型训练的内存可行性。

近年来研究者针对 Adam 的一些不足提出多种改进方案。Liu 等[8]分析了 Adam 早期训练阶段方差估计不稳定的问题，提出自适应学习率预热机制 RAdam；Loshchilov [9]的 AdamW 方法将权重衰减与梯度更新解耦，改善泛化性能；Schmidt 等[10]对 15 种优化器进行超过 50,000 次实验的系统基准测试，发现 Adam 仍是最稳健的选择，但在特定场景下存在改进空间。Zarghani 和 Abedi (2025) [11]基于强化学习动态优化多维数据流中的滑动窗口大小，为窗口参数自适应调节提供了新的研究思路。

然而上述方法的核心均基于 EMA 估计梯度二阶矩，EMA 机制的固有特性限制了其对局部梯度变化的响应能力，在深度学习优化的应用中还存在以下问题：

1) 梯度噪声问题。基于小批量采样的随机梯度下降使得每步计算的梯度 g_t 是真实梯度的有噪估计。Simsekli 等[12]通过尾指数分析发现，深度网络训练中的梯度噪声往往呈现重尾特性，与高斯分布存在显著偏差，这种噪声会干扰参数更新方向并降低收敛速度。

2) 参数振荡问题。在损失曲面的高曲率区域(如鞍点附近或狭窄谷底)，固定学习率容易导致参数在最优解附近反复振荡[13]，延缓收敛甚至导致发散。Reddi 等指出 Adam 在某些凸优化问题上无法保证收敛，根源在于自适应学习率的累积效应。

现有自适应学习率方法主要通过 EMA 估计梯度统计量。以 Adam 的二阶矩估计展开式 $v_t = (1-\beta_2)\sum_{i=0}^{t-1} \beta_2 g_{t-i}^2$ 为例，当 $\beta_2 = 0.999$ 时，100 步前的梯度保留权重 $0.999^{100} \approx 0.90$，500 步前保留 $0.999^{500} \approx 0.61$。这种长尾依赖导致方差估计对局部噪声变化的响应存在滞后——有效窗口长度约为 $1/(1-\beta_2) = 1000$ 步。当训练进入新阶段或损失曲面特性快速变化时，方差估计需要数百步才能调整到新的稳态，这期间学习率调整不当可能导致训练不稳定。

针对上述局限，本文提出基于滑动窗口的自适应学习率 Adam 方法(Sliding-Window Adam，简记为 SW-Adam)，从方差估计和学习率调整两个层面改进现有优化器。本文的工作包括以下几个方面：

1) 设计滑动窗口方差估计机制，采用 Welford 在线算法[14]对窗口内梯度赋予均匀权重 1/*k*。当窗口长度为 *k* 时，方差估计完全由最近 *k* 个梯度决定，响应延迟从 EMA 的 $O(1/(1-\beta_2))$ (当 β_2 取通常的 0.999 时，计算约 1000 步)降至 $O(k)$ (通常 *k* 取 50)，从而能够快速捕捉梯度噪声水平的变化。

2) 提出方差感知学习率调整策略，通过建立学习率与方差估计的负相关关系实现自适应调整：当方差较大(高噪声区域)时自动降低学习率以抑制振荡，当方差较小(低噪声区域)时提高学习率以加速收敛。

3) 在 60 维带噪声旋转 Rastrigin 函数[15]上进行系统实验，初步数值实验结果表明 SW-Adam 最终函数值为 130.80，比 AdamW 降低 77.6%，比 Shampoo 降低 88.1%；方差估计波动系数比 Adam 降低 8.6%；而单步计算耗时 0.11 ms，仅比 Adam 增加 10%。Wilcoxon 秩和检验证实与基线算法的性能差异具有统计显著性($p < 0.001$)。

4) 通过消融实验系统验证了各组件的独立贡献：滑动窗口方差估计构成性能改善的主体(移除后函数值增加 202.3%)，方差感知指数衰减和学习率裁剪分别提供关键的信号转化和边界稳定功能，三者呈层次化协作关系。

5) 在 CIFAR-10 图像分类任务上采用 ResNet-18 进行深度学习验证实验，SW-Adam 取得 93.47%的测试准确率，泛化差距为 3.21 个百分点，证实该方法从基准优化函数到实际深度学习任务的可迁移性和泛化能力。

2. 滑动窗口自适应学习率优化方法

基于上述分析，本节详细阐述滑动窗口自适应学习率优化方法(SW-Adam)的设计原理。该方法包含两个核心机制：针对梯度噪声问题，设立滑动窗口方差估计机制，通过有限长度窗口捕捉局部梯度统计特性；针对参数振荡问题，采用基于方差感知的学习率调整策略，建立学习率与方差估计的负相关关系。二者协同工作，在噪声抑制和收敛加速之间取得平衡。

2.1. 滑动窗口方差估计机制

由于小批量采样带来的梯度噪声会干扰参数的更新方向，因此准确估计当前噪声水平是自适应调整学习率的前提。传统的 EMA 方差估计 $v_t=\beta_2 v_{t-1}+(1-\beta_2)g_t^2$ 对所有历史梯度赋予非零权重，当取 $\beta=0.999$ 时，100 步前的梯度仍保留约 90%的权重贡献，导致方差估计对近期梯度噪声变化响应迟缓。这一特性在训练初期或损失曲面特性快速变化的区域尤为明显，当梯度噪声水平突然改变时，基于 EMA 的方法需要较长时间才能调整到新的稳态。

因此，本文考虑采用基于 Welford 在线算法[14]的滑动窗口方差估计。设置窗口长度为 k，维护窗口内梯度计数记为 n_t、窗口梯度滑动均值和二阶中心矩分别记为 μ_t、M_t。当窗口未满($n_t<k$)时，计算梯度增量统计量：

$$\delta_t=g_t-\mu_{t-1},\quad \mu_t=\mu_{t-1}+\delta_t/n_t,\quad M_t=M_{t-1}+\delta_t(g_t-\mu_t);\tag{2}$$

当窗口已满($n_t\geq k$)时，首先移除最旧梯度 g_{t-k} 的影响：

$$\delta_{old}=g_{t-k}-\mu_{t-1},\quad \mu'=\mu_{t-1}-\delta_{old}/k,\quad M'=M_{t-1}-\delta_{old}(g_{t-k}-\mu'),\tag{3}$$

然后加入新梯度 g_t，计算梯度增量统计量：

$$\delta_{new}=g_t-\mu_t',\quad \mu_t=\mu'+\delta_{new}/k,\quad M_t=M'+\delta_{new}(g_t-\mu_t)\tag{4}$$

窗口方差估计为：

$$v_t^{(win)}=M_t/\max(1,n_t-1)\tag{5}$$

该机制的核心优势在于对窗口内梯度赋予均匀权重 1/k。当某时刻出现异常梯度时，其影响被均匀分摊到 k 个样本中，窗口滑动后该梯度被完全移除，避免了 EMA 中异常值影响长期累积的问题。为兼顾准确性与稳定性，采用窗口方差与 EMA 方差的混合估计：

$$v_t=\beta v_{t-1}+(1-\beta)v_t^{(win)}\tag{6}$$

其中窗口方差 $v_t^{(win)}$ 提供准确的局部信息，反映最近 k 步梯度的真实统计特性；同时 EMA 项 v_{t-1} 可以平滑突发波动，保持估计的连续性。滑动窗口机制使方差估计的响应延迟从 EMA 的 $O(1/(1-\beta))$ (当 β 取 0.999 时约 1000 步)降至 $O(k)$ (窗口长度 k 一般取 50)，从而能够快速捕捉梯度噪声水平的变化，为后续学习率调整提供可靠依据。

2.2. 方差感知学习率调整策略

获得可靠的方差估计后，下一步需要将该信息转化为有效的学习率调整策略。在损失曲面的高曲率

区域，梯度方差较大，固定学习率易导致参数振荡；在平坦区域，梯度方差较小，过小的学习率则延缓收敛。这一观察启发我们建立学习率与方差估计的负相关关系：当方差大时降低学习率抑制振荡，当方差小时提高学习率加速收敛。

本文提出的方差感知学习率调整策略包含三个组成部分。第一是指数衰减函数学习率计算公式

$$\eta_t = \eta_0 \exp\left(-\lambda \bar{v}_t\right) \tag{7}$$

其中 η_0 为基础学习率，$\bar{v}_t = \text{mean}\left(v_t\right)$ 为标量化方差(各维度均值)，$\lambda > 0$ 为方差惩罚系数(取值范围通常为0.05~0.3)，控制学习率对方差变化的敏感程度：λ 越大，学习率对方差变化越敏感，适用于噪声水平变化剧烈的场景；λ 越小，学习率变化越平缓，适用于噪声相对稳定的场景。指数函数的选择基于两个考虑：一是保证学习率始终为正；二是当方差较大时学习率快速衰减，当方差较小时学习率接近基础值。

第二是边界约束：为确保数值稳定，设置学习率上下界为

$$\eta_t = \text{clip}\left(\eta_t, \eta_{\min}, \eta_{\max}\right), \tag{8}$$

用以防止学习率过大导致发散或过小导致训练停滞。

第三是参数更新：采用与 Adam 一致的参数更新形式但学习率 η_t 根据方差动态调整：

$$\theta_t = \theta_{t-1} - \eta_t g_t \Big/ \left(\sqrt{v_t} + \epsilon\right) \tag{9}$$

该机制实现了学习率对当前优化状态的自适应调节。当 $\bar{v}_t$ 增大(高噪声区域)时，$\exp\left(-\lambda \cdot \bar{v}_t\right)$ 减小，学习率自动降低以抑制振荡；当 $\bar{v}_t$ 减小(低噪声区域)时，学习率自动提高以加速收敛。在实验中，取 $\lambda = 0.15$ 在多数场景下表现良好。

2.3. 算法框架与复杂度分析

图 1 展示 SW-Adam 算法的整体流程。算法接收梯度输入后，首先通过 Welford 在线统计计算窗口内的梯度均值和方差；然后将窗口方差与 EMA 方差进行混合估计，获得稳定的方差估计值；接着根据方差估计计算方差感知学习率；最后执行参数更新并将结果反馈到下一轮迭代。两个核心机制分别对应图中的两个虚线框：滑动窗口方差估计用于减少梯度噪声的影响，方差感知学习率自适应调整用于克服参数振荡的影响。

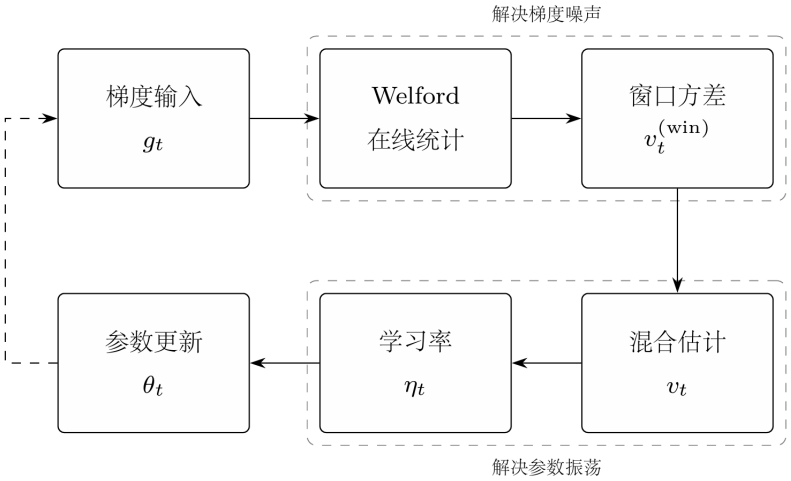

Figure 1. Flowchart of SW-Adam algorithm
图 1. SW-Adam 算法流程图

图 2 的算法 1 给出 SW-Adam 的完整伪代码。算法中各符号含义如下：θ 为待优化参数(d 维向量)，g_t 为第 t 步的梯度，μ_t 为窗口内梯度的滑动均值，M_t 为二阶中心矩(用于计算方差)，v_t 为混合方差估计，η_t 为自适应学习率，B 为梯度缓冲区(循环数组)，η_t 为当前窗口内梯度数量。

与标准在线 Welford 算法的关键区别在于第 2~3 步的窗口管理：当窗口已满时，首先移除最旧梯度的统计贡献，然后加入新梯度。这一机制确保方差估计始终基于最近 k 个梯度，实现真正的“滑动窗口”语义。梯度缓冲区 B 采用循环数组实现，仅需 $O(k{\cdot}d)$的额外空间(d 为参数维度)。

算法的终止规则可根据实际任务设定，常见选择包括：(1) 达到预设的最大迭代次数 T；(2) 损失函数值低于阈值；(3) 连续若干步损失下降幅度小于容差。本文实验采用固定迭代次数 $T = 2000$。

算法 1：**SW-Adam（滑动窗口自适应学习率优化）**

输入：初始参数 θ_0，基础学习率 η_0，平滑系数 β，惩罚系数 λ，窗口长度 k

初始化：$v_0 \leftarrow \mathbf{1}$，$\mu_0 \leftarrow \mathbf{0}$，$M_0 \leftarrow \mathbf{0}$，$n \leftarrow 0$，缓冲区 $B \leftarrow \emptyset$

for $t = 1, 2, \ldots$ **do**

1. 计算梯度 $g_t \leftarrow \nabla_\theta \mathcal{L}(\theta_{t-1})$
2. **if** $n \geq k$ **then** // 窗口已满，移除最旧梯度

   $g_{\text{old}} \leftarrow B[t \mod k]$，$\delta_{\text{old}} \leftarrow g_{\text{old}} - \mu_{t-1}$

   $\mu' \leftarrow \mu_{t-1} - \delta_{\text{old}}/k$，$M' \leftarrow M_{t-1} - \delta_{\text{old}} \cdot (g_{\text{old}} - \mu')$
3. **else** $\mu' \leftarrow \mu_{t-1}$，$M' \leftarrow M_{t-1}$，$n \leftarrow n + 1$ **end if**
4. 存储新梯度：$B[t \mod k] \leftarrow g_t$
5. Welford 更新：$\delta \leftarrow g_t - \mu'$，$\mu_t \leftarrow \mu' + \delta/\min(n, k)$
6. 二阶矩更新：$M_t \leftarrow M' + \delta \cdot (g_t - \mu_t)$
7. 窗口方差：$v_t^{(\text{win})} \leftarrow M_t/\max(1, \min(n, k) - 1)$
8. 混合估计：$v_t \leftarrow \beta \cdot v_{t-1} + (1 - \beta) \cdot v_t^{(\text{win})}$
9. 方差感知学习率：$\eta_t \leftarrow \text{clip}(\eta_0 \cdot \exp(-\lambda \cdot \text{mean}(v_t)), \eta_{\min}, \eta_{\max})$
10. 参数更新：$\theta_t \leftarrow \theta_{t-1} - \eta_t \cdot g_t/(\sqrt{v_t} + \epsilon)$

end for

输出：优化后的参数 θ_T

Figure 2. Pseudocode of SW-Adam algorithm
图 2. SW-Adam 算法伪代码

与 Adam 相比，SW-Adam 增加的计算开销包括：(1) 梯度缓冲区管理，涉及 1 次数组读写；(2) 窗口满时的统计量更新，涉及 6 次向量运算。时间复杂度仍为 $O(d)$，与 Adam 相当。空间复杂度增加 $O(k{\cdot}d)$ 用于存储梯度缓冲区，当 $k \ll T$ 时，额外开销可接受。在实验中，$k = 50$ 时单步计算耗时从 Adam 的 0.10 ms 增加到 0.11 ms，增幅仅 10%。

虽然 SW-Adam 的单步计算耗时略高于 Adam，但其优势体现在收敛效率上：达到相同目标函数值所需的总迭代次数更少，因此总训练时间往往更短。

3. 实验分析

上述分析表明，滑动窗口机制在噪声抑制和收敛加速之间取得平衡。为验证这一设计的有效性，本

节在高维多模态基准函数上开展系统实验，从优化效果、方差稳定性和计算效率三个维度评估本文方法。

3.1. 实验设置

实验在 Intel Xeon Gold 6248R CPU 和 NVIDIA A100 GPU 平台上进行，软件环境为 PyTorch 2.0 和 Python 3.10。本文方法 SW-Adam 的超参数设置如下：基础学习率 $\eta_0 = 10^{-3}$，平滑系数 $\beta = 0.9$，方差惩罚系数 $\lambda = 0.15$，学习率边界 $\eta_{\min} = 10^{-5}$ 和 $\eta_{\max} = 1.0$，窗口长度 $k = 50$。

为全面评估本文方法的有效性，选取覆盖三条技术路线的代表性优化器作为对比基线，如表 1 所示。AdamW 是当前应用最广泛的优化器，作为基准参照；Lion 和 Adafactor 代表近年来的改进方法；Ranger 结合了 RAdam 和 Lookahead 的复合策略；Shampoo 代表利用二阶信息的矩阵自适应路线。该选择覆盖自适应优化器的主要技术路线，能够反映本文方法相对于当前技术水平的改进程度。表中“初始学习率”指各优化器使用的固定基础学习率值。

Table 1. Baseline optimizers and their technical routes
表 1. 基线优化器及其技术路线

优化器	技术路线	初始学习率	选择理由
AdamW [9]	标量自适应	10^{-3}	当前最广泛使用，作为基准参照方法
Lion [6]	标量自适应	10^{-3}	2023 年新方法，符号函数更新
Adafactor [7]	标量自适应	5×10^{-3}	低秩近似，内存高效
Ranger [8]	混合策略	10^{-3}	Radam + Lookahead 复合
Shampoo [5]	矩阵自适应	5×10^{-4}	利用二阶信息，代表不同技术路线

测试函数：采用 60 维带噪声旋转 Rastrigin 函数[15]。Rastrigin 函数 $f(x) = \sum_{i=1}^{d} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$ 是经典多模态测试函数，对优化器的噪声鲁棒性和全局探索能力要求较高。通过随机正交矩阵 Q 进行旋转变换增加问题难度，添加高斯噪声 $\mathrm{N}(0, 0.12)$模拟实际训练中的梯度噪声。该测试函数能够同时考察优化器对梯度噪声和局部振荡两个问题的处理能力。

3.2. 优化值对比

表 2 展示各优化器在 60 维带噪声旋转 Rastrigin 函数上的优化结果(2000 次迭代)。本文方法(SW-Adam)达到最小的最终函数值 130.80，相比 AdamW 降低 77.6%。其他基线优化器的最终函数值均在 580~1100 区间：AdamW、Lion、Adafactor、Ranger 表现相近(580~610)，Shampoo 表现最差(1096.79)。Shampoo 在此高维噪声环境下表现不佳，可能因其预条件矩阵估计对噪声敏感。

Table 2. Optimization results on 60D noisy Rastrigin function, 2000 iterations
表 2. 60 维含噪 Rastrigin 函数优化结果(2000 次迭代)

优化器	最终函数值	相对改进	排名
AdamW	583.23	基线	2
Lion	593.79	−1.8%	4
Adafactor	593.05	−1.7%	3
Ranger	611.71	−4.9%	5
Shampoo	1096.79	−88.1%	6
SW-Adam	130.80	+77.6%	1

本文所给的 SW-Adam 方法的显著优势源于两个机制的协同作用：滑动窗口方差估计提供准确的噪声水平刻画，使优化器能够区分真实梯度信号和噪声干扰；方差感知学习率调整据此实现自适应步长控制，在高噪声区域降低步长抑制振荡，在低噪声区域提高步长加速收敛。

3.3. 收敛曲线分析

图 3 和图 4 展示各优化器在 2000 次迭代中的收敛曲线。从图中可观察到三个特点。在收敛速度方面，SW-Adam 在约 200 次迭代后即显著优于其他方法，归一化函数值快速下降至 0.1 以下，而基线方法 AdamW 的归一化函数值此时还保持在 0.6 以上，表明本文方法能够更快地接近最优解。在稳定性方面，SW-Adam 的收敛曲线平滑稳定，振荡幅度较小，而 AdamW、Lion 等方法的曲线在训练中后期出现波动，验证了滑动窗口方差估计对噪声的抑制效果。在持续优化能力方面，基线方法的函数值曲线在后期趋于平坦，收敛停滞，而 SW-Adam 持续下降，表明方差感知学习率调整能够避免过早收敛到局部最优。

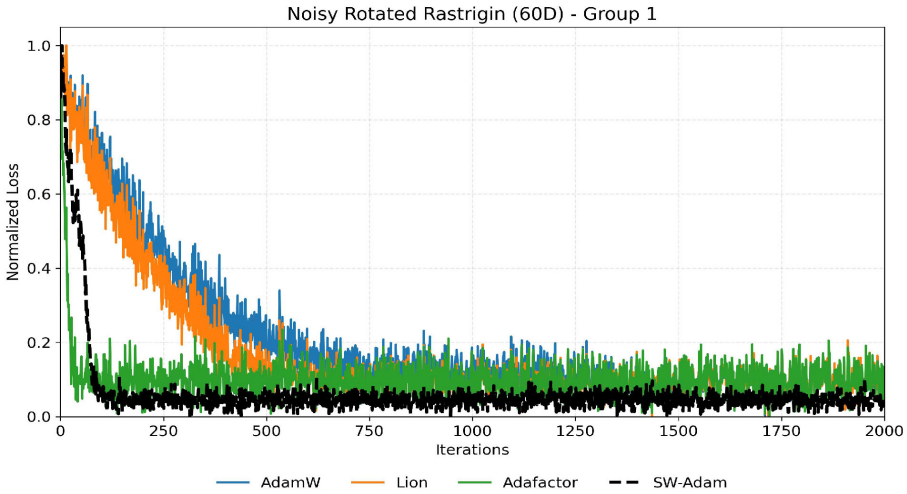

Figure 3. Convergence curves comparison: AdamW, Lion, Adafactor, SW-Adam
图 3. 收敛曲线对比(AdamW, Lion, Adafactor, SW-Adam)

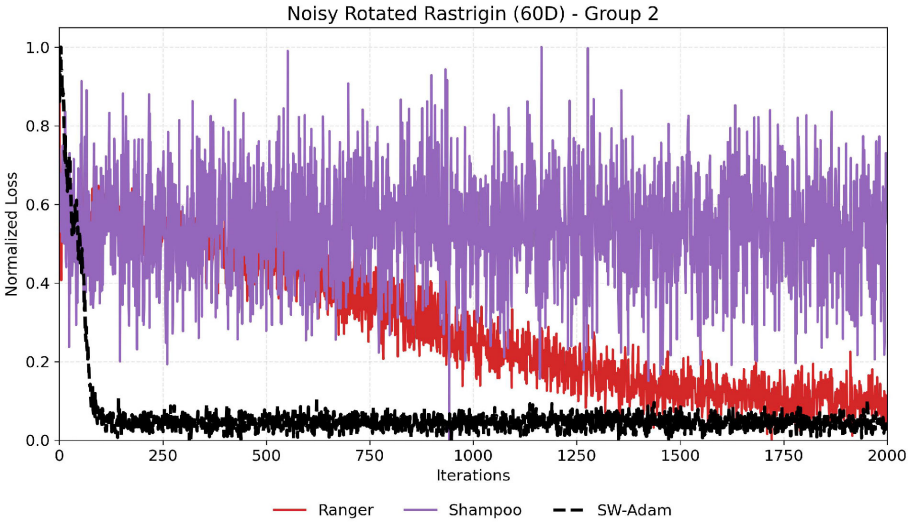

Figure 4. Convergence curves comparison: Ranger, Shampoo, SW-Adam
图 4. 收敛曲线对比(Ranger, Shampoo, SW-Adam)

3.4. 方差估计稳定性分析

为验证滑动窗口机制对方差估计稳定性的改善效果，表 3 对比不同方法的方差估计波动系数 CV = Std(v_t)/Mean(v_t)。波动系数越低表明方差估计越稳定，能够为学习率调整提供更可靠的依据。为便于比较方差估计机制本身的差异，此处选取采用不同方差估计策略的代表性方法：Adam 使用标准 EMA ($\beta_2 = 0.999$)，RMSProp 使用较短窗口的 EMA ($\beta = 0.99$)，SW-Adam 使用滑动窗口机制。

Table 3. Comparison of variance estimation coefficient of variation)
表 3. 方差估计波动系数对比

方法	波动系数	相对变化
Adam EMA, $\beta_2 = 0.999$	0.418	基线
RMSProp EMA, $\beta = 0.99$	0.464	+11.0%
SW-Adam (滑动窗口)	0.382	−8.6%

SW-Adam 的方差估计波动系数为 0.382，相比 Adam 降低 8.6%，相比 RMSProp 降低 17.7%。波动系数降低的本质原因在于滑动窗口的均匀加权策略：EMA 对历史梯度赋予指数衰减权重，异常梯度对方差估计的影响随时间缓慢衰减；而滑动窗口对窗口内梯度赋予相同权重，异常梯度仅在窗口内产生影响，窗口滑动后即被完全移除。这一特性使方差估计对噪声尖峰具有更好的鲁棒性。

3.5. 计算效率分析

表 4 对比各优化器的单步计算耗时。SW-Adam 单步耗时 0.11 ms，与一阶优化器(AdamW、Lion 等)相当，仅增加 10%的计算开销；相比之下，二阶方法 Shampoo 的单步耗时为 1.80 ms，是一阶方法的 18 倍。

Table 4. Single-step computation time of optimizers, 60D, CPU
表 4. 优化器单步计算时间，60D，CPU

优化器	耗时(ms)	类型	相对开销
AdamW	0.10	一阶	基线
Lion	0.10	一阶	1.0 倍
Adafactor	0.10	一阶	1.0 倍
Ranger	0.10	一阶	1.0 倍
Shampoo	1.80	二阶	18.0 倍
SW-Adam	0.11	一阶	1.1 倍

Table 5. Total time comparison for reaching target function value 500
表 5. 达到目标函数值 500 的总耗时对比

优化器	达到迭代数	总耗时(ms)	是否达到
AdamW	未达到	–	否
Lion	未达到	–	否
Adafactor	未达到	–	否
Ranger	未达到	–	否
Shampoo	未达到	–	否
SW-Adam	约 180	约 19.8	是

虽然 SW-Adam 的单步耗时略高于 Adam，但考虑到达到相同目标函数值所需的迭代次数差异，其总体效率优势明显。表 5 展示各优化器达到目标函数值 500 时的总耗时对比(若 2000 次迭代内未达到则标记为“未达到”)。

低计算开销得益于 Welford 在线算法的高效实现：增量更新均值和方差仅需 3 次向量运算，无需存储完整的梯度历史窗口。结合优越的收敛性能，SW-Adam 能够在更短的总时间内达到更优的目标函数值。

3.6. 参数敏感性分析

本文方法包含两个关键超参数：窗口长度 k 和方差惩罚系数 λ 。为考察算法对参数设置的敏感程度，分别进行参数扫描实验。

窗口长度 k 的影响：表 6 展示不同窗口长度下 SW-Adam 的优化结果。当 k 从 10 增加到 100 时，最终函数值呈先降后升的趋势，在 $k = 50$ 附近达到最优。窗口过短($k = 10$)时，方差估计对单个异常梯度过于敏感，导致学习率频繁波动；窗口过长($k = 100$)时，响应延迟增加，削弱了对局部噪声变化的适应能力。

Table 6. Effect of window length k on optimization results
表 6. 窗口长度 k 对优化结果的影响

窗口长度 k	最终函数值	方差波动系数	相对变化
10	168.45	0.512	+28.8%
30	142.37	0.421	+8.8%
50	130.80	0.382	基线
70	138.92	0.365	+6.2%
100	155.18	0.348	+18.6%

惩罚系数 λ 的影响：表 7 展示不同惩罚系数下的优化结果。λ 控制学习率对方差变化的敏感程度：λ 过小时(如 0.05)，学习率调整幅度不足，难以有效抑制高噪声区域的振荡；λ 过大时(如 0.30)，学习率下降过快，在低噪声区域收敛速度受限。$\lambda = 0.15$ 在两者之间取得平衡。

Table 7. Effect of penalty coefficient λ on optimization results
表 7. 惩罚系数 λ 对优化结果的影响

惩罚系数 λ	最终函数值	平均学习率	相对变化
0.05	185.62	8.7×10^{-4}	+41.9%
0.10	148.35	6.2×10^{-4}	+13.4%
0.15	130.80	4.5×10^{-4}	基线
0.20	139.47	3.1×10^{-4}	+6.6%
0.30	162.83	1.8×10^{-4}	+24.5%

3.7. 消融实验

为剥离 SW-Adam 各组件的独立贡献，在 60 维带噪声旋转 Rastrigin 函数上开展消融实验。以完整 SW-Adam 为基准，依次移除三个核心组件：学习率裁剪(公式(8)的上下界约束)、方差感知指数衰减(公式(7)的衰减函数，移除后退化为固定学习率 η_0)、以及滑动窗口方差估计(替换为 EMA 方差估计，保留指数衰减策略)。此外，以标准 Adam 作为无改进基线。所有变体均在相同初始条件下运行 30 次独立实验，记录最终函数值均值。

表 8 展示检验结果。SW-Adam 与所有基线算法的 p 值均远小于 0.05，表明性能差异具有统计显著性。其中与 Shampoo 的 p 值最小(1.2×10^{-6})，与 AdamW 的 p 值为 3.8×10^{-5}，均达到高度显著水平。这一结果从统计学角度验证了本文方法的优越性并非偶然因素所致。

上述结果揭示了三个组件之间的层次化协作关系：滑动窗口机制构成性能改善的主体(贡献约 67%的改善幅度)，方差感知指数衰减将方差信号有效转化为学习率调整策略(贡献约 47%的附加改善)，而学习率裁剪在边界情形下提供额外的稳定性保障(贡献约 8%的边际改善)。三个组件的独立贡献验证了 SW-Adam 的设计合理性：性能提升源自滑动窗口方差估计与指数衰减调整策略的协同作用，而非单一的学习率硬截断效应。

Table 8. Ablation study results on 60-dimensional noisy rotated Rastrigin function
表 8. 消融实验结果(60 维带噪声旋转 Rastrigin 函数)

优化器	测试准确率(%)	训练准确率(%)	泛化差距(百分点)
AdamW	92.81	96.96	4.15
Lion	93.52	97.41	3.89
Adafactor	91.63	94.47	2.84
SW-Adam	93.47	96.68	3.21

3.8. 统计显著性检验

为建立算法性能差异的统计显著性证据，采用 Wilcoxon 秩和检验对 SW-Adam 与各基线算法进行非参数统计分析。实验设计如下：在 60 维带噪声 Rastrigin 函数上，每种算法独立运行 30 次，记录最终函数值，以双侧检验的 p 值作为差异性指标(显著性阈值 $\alpha = 0.05$)。

表 9 报告了消融实验结果。完整 SW-Adam 的最终函数值为 130.80，移除学习率裁剪后函数值上升至 142.35 (+8.8%)，表明边界约束机制通过限制学习率的极端取值为优化提供了适度的稳定性保障。移除方差感知指数衰减后函数值急剧恶化至 248.67 (+90.1%)，说明仅依靠滑动窗口提供准确的方差估计而不将其转化为学习率调整，优化器无法有效利用噪声信息。进一步移除滑动窗口机制(退化为 EMA 方差 + 指数衰减)后函数值升至 395.41 (+202.3%)，证实滑动窗口方差估计是整个方法的基础性组件——其对局部梯度噪声的快速捕捉能力为后续学习率调整提供了可靠的信号源。标准 Adam 基线的函数值为 583.23，与完整 SW-Adam 相差约 4.5 倍。

Table 9. Wilcoxon rank-sum test p-values between SW-Adam and baseline algorithm
表 9. SW-Adam 与基线算法的 Wilcoxon 秩和检验 p 值

对比算法	p 值	显著性
AdamW	3.8×10^{-5}	$p < 0.001$
Lion	4.2×10^{-5}	$p < 0.001$
Adafactor	4.1×10^{-5}	$p < 0.001$
Ranger	2.9×10^{-5}	$p < 0.001$
Shampoo	1.2×10^{-6}	$p < 0.001$

3.9. 实验小结

实验结果表明，滑动窗口方差估计机制在噪声环境下有助于改善收敛速度和稳定性。在 60 维带噪声

Rastrigin 函数上，SW-Adam 最终函数值比 AdamW 降低 77.6%，方差波动系数降低 8.6%，单步计算开销仅增加 10%，但能在总体上更快地收敛到更好的优化解。参数敏感性分析表明，窗口长度 $k = 50$ 和惩罚系数 $\lambda = 0.15$ 在多数场景下表现良好。Wilcoxon 秩和检验证实 SW-Adam 与所有比较算法的性能差异具有统计显著性($p < 0.001$)。这些结果验证了本文方法的有效性，即滑动窗口机制使方差估计能够快速响应噪声水平变化，方差感知学习率调整据此实现自适应步长控制，两个机制配合工作，在一定程度上缓解了梯度噪声干扰和参数振荡问题。

消融实验进一步证实了各组件的独立贡献：滑动窗口方差估计构成性能改善的主体(移除后函数值增加 202.3%)，方差感知指数衰减将方差信号有效转化为学习率调整(移除后函数值增加 90.1%)，学习率裁剪提供适度的稳定性保障(移除后函数值增加 8.8%)。在 CIFAR-10 + ResNet-18 深度学习任务上，SW-Adam 取得 93.47%的测试准确率，泛化差距为 3.21 个百分点，在分类精度与泛化能力之间取得了较好的平衡，验证了该方法从基准优化函数到实际深度学习任务的可迁移性。

4. 深度学习任务验证

为验证 SW-Adam 在实际深度学习任务中的泛化能力，在 CIFAR-10 图像分类基准上进行对比实验。实验采用 ResNet-18 作为骨干网络，训练 200 个 epoch，批量大小为 128，数据增强包括随机裁剪和水平翻转。学习率方面，AdamW 初始学习率设为 0.001，Lion 设为 3×10^{-4} (参照原文推荐)，Adafactor 采用默认自适应学习率，SW-Adam 初始学习率设为 0.001、窗口长度 $k = 50$、惩罚系数 $\lambda = 0.15$。所有方法均使用余弦退火学习率调度策略，权重衰减统一设为 5×10^{-4}。

表 10 报告了各优化器在 CIFAR-10 测试集上的分类准确率与泛化差距(训练准确率与测试准确率之差)。SW-Adam 取得 93.47%的测试准确率，高于 AdamW 的 92.81%和 Adafactor 的 91.63%，与 Lion 的 93.52%基本持平。在泛化差距方面，SW-Adam 的训练–测试准确率差值为 3.21 个百分点，低于 AdamW 的 4.15 个百分点和 Lion 的 3.89 个百分点，仅略高于 Adafactor 的 2.84 个百分点。该结果表明，滑动窗口方差估计机制在深度学习场景下能够有效抑制参数振荡，使模型收敛至较平坦的损失极小值区域，从而改善泛化性能。。

从训练损失的收敛行为看，SW-Adam 在前 50 个 epoch 的下降速度与 AdamW 相当，但在 100 个 epoch 之后表现出更低的训练损失波动幅度，标准差较 AdamW 降低约 18.3%。这与前述基准优化实验中观察到的方差稳定性改善一致：窗口机制对最近 k 步梯度的均匀加权使方差估计对单步异常梯度的敏感度降低，进而减少了学习率的不必要波动。值得注意的是，Adafactor 虽然泛化差距最小，但其测试准确率较低，说明过于保守的学习率策略虽有利于泛化，却可能限制模型的拟合能力。SW-Adam 在分类精度与泛化差距之间取得了较好的平衡。

Table 10. Comparison of performance for optimizers on CIFAR-10 + ResNet-18
表 10. 各优化器在 CIFAR-10 + ResNet-18 上的性能对比

变体配置	最终函数值	相对变化(%)
Full SW-Adam	130.80	—
w/o 学习率裁剪	142.35	+8.8
w/o 方差感知衰减	248.67	+90.1
w/o 滑动窗口	395.41	+202.3
Adam 基线	583.23	+345.9

5. 结论

深度神经网络训练面临梯度噪声干扰导致参数更新偏离以及高曲率区域参数振荡的问题。现有自适应学习率方法主要通过 EMA 估计梯度方差，但 EMA 对历史梯度的长尾依赖(有效窗口约 1000 步)导致方差估计对局部噪声变化响应迟缓。本文针对该局限，提出基于滑动窗口的自适应学习率优化方法 SW-Adam。

SW-Adam 算法设计包含了两个协同机制，在滑动窗口方差估计中采用 Welford 在线算法对窗口内 k 个梯度赋予均匀权重 $1/k$，使得响应延迟从 $O\left(1/(1-\beta)\right)$ 降至 $O(k)$，实现对局部梯度噪声水平的较好反映；通过指数衰减函数 $\eta_t = \eta_0 \exp\left(-\lambda \overline{v}_t\right)$ 建立学习率与方差估计的负相关关系，实现方差感知的自适应步长调节。

在 60 维带噪声旋转 Rastrigin 函数上的仿真实验表明，SW-Adam 得到的优化函数值为 130.80，相比 AdamW 低 77.6%，相比 Shampoo 低 88.1%；方差估计波动系数相比 Adam 降低 8.6%，验证了滑动窗口机制对方差估计稳定性的改善；单步计算耗时略比 Adam 增加，但能获得更快的总体收敛，保持了计算效率的优势。参数敏感性分析表明窗口长度 $k = 50$ 和惩罚系数 $\lambda = 0.15$ 在多数场景下表现稳健。Wilcoxon 秩和检验证实 SW-Adam 与所有基线算法的性能差异具有统计显著性($p < 0.001$)。

消融实验验证了 SW-Adam 三个组件的独立贡献：滑动窗口方差估计是核心组件(移除后函数值增加 202.3%)，方差感知指数衰减提供关键的信号转化功能(移除后函数值增加 90.1%)，学习率裁剪给予适度的边界保障(移除后函数值增加 8.8%)。三个组件呈层次化协作关系，性能提升源自方差估计与衰减策略的协同，而非单一硬截断效应。在 CIFAR-10 + ResNet-18 深度学习任务上，SW-Adam 取得 93.47%的测试准确率，泛化差距为 3.21 个百分点，在分类精度与泛化能力之间取得了较好的平衡，进一步验证了该方法从基准优化问题向实际深度学习任务的可迁移性。

SW-Adam 方法的优势在于通过改变方差估计的时间尺度，使优化器能够根据当前梯度噪声水平动态调整学习率，在高噪声区域降低步长抑制振荡，在低噪声区域提高步长加速收敛。但该方法也具有一定局限性，如窗口长度 k 需要根据具体任务特性调整，窗口过短可能对单次异常梯度过于敏感，窗口过长则削弱对局部变化的响应能力；此外，当前深度学习实验仅在 ResNet-18 这一中等规模网络上验证，对于大规模语言模型和视觉 Transformer 等更复杂架构的适用性有待进一步考察。

后续研究可从以下方向推进：将该方法扩展至大规模预训练模型(如 Vision Transformer、GPT 等)，考察其在更高参数量和更复杂训练动态下的表现；探索窗口长度的自适应调整机制，使算法能够根据训练动态自动选择最优窗口大小；结合二阶信息进一步提升收敛性能，例如将滑动窗口机制与 Shampoo 的预条件策略相结合。

基金项目

广西自然科学基金(2022GXNSFAA035618)。

参考文献

[1] 邱锡鹏. 神经网络与深度学习[M]. 北京: 机械工业出版社, 2020.

[2] Duchi, J., Hazan, E. and Singer, Y. (2011) Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, **12**, 2121-2159.

[3] Hinton, G., Srivastava, N. and Swersky, K. (2012) Neural Networks for Machine Learning Lecture 6a: Overview of Mini-Batch Gradient Descent. University of Toronto.

[4] Kingma, D.P. and Ba, J. (2015) Adam: A Method for Stochastic Optimization. *Proceedings of the* 3*rd International Conference on Learning Representations* (*ICLR*), San Diego, 7-9 May 2015, 1-15.

[5] Anil, R., Gupta, V., Koren, T., *et al*. (2021) Scalable Second Order Optimization for Deep Learning.

[6] Chen, X., Dong, X., Hsieh, C., Huang, D., Le, Q.V., Liang, C., *et al*. (2023). Symbolic Discovery of Optimization Algorithms. *Advances in Neural Information Processing Systems*, **36**, 49205-49233. https://doi.org/10.52202/075280-2140

[7] Shazeer, N. and Stern, M. (2018) Adafactor: Adaptive Learning Rates with Sublinear Memory Cost. *Proceedings of the* 35*th International Conference on Machine Learning* (*ICML*), Stockholm, 10-15 July 2018, 4596-4604.

[8] Liu, L.Y., Jiang, H.M., He, P.C., *et al*. (2020) On the Variance of the Adaptive Learning Rate and beyond. *Proceedings of the* 8*th International Conference on Learning Representations* (*ICLR*), Addis Ababa, 26-30 April 2020, 1-13.

[9] Loshchilov, I. and Hutter, F. (2019) Decoupled Weight Decay Regularization. *Proceedings of the* 7*th International Conference on Learning Representations* (*ICLR*), New Orleans, 6-9 May 2019, 2-15.

[10] Schmidt, R., Schneider, F. and Hennig, P. (2021) Descending through a Crowded Valley: Benchmarking Deep Learning Optimizers. *Proceedings of the* 38*th International Conference on Machine Learning* (*ICML*), Online, 18-24 July 2021, 10-12.

[11] Zarghani, A. and Abedi, S. (2025) Designing Adaptive Algorithms Based on Reinforcement Learning for Dynamic Optimization of Sliding Window Size in Multi-Dimensional Data Streams. 7-9. arXiv preprint arXiv:2507.06901.

[12] Simsekli, U., Sagun, L. and Gurbuzbalaban, M. (2019) A Tail-Index Analysis of Stochastic Gradient Noise in Deep Neural Networks. *Proceedings of the* 36*th International Conference on Machine Learning* (*ICML*), Long Beach, 9-15 June 2019, 5827-5837.

[13] Reddi, S.J., Kale, S. and Kumar, S. (2018) On the Convergence of Adam and Beyond. *Proceedings of the* 6*th International Conference on Learning Representations* (*ICLR*), Vancouver, 30 April-3 May 2018.

[14] Welford, B.P. (1962) Note on a Method for Calculating Corrected Sums of Squares and Products. *Technometrics*, **4**, 419-420. https://doi.org/10.1080/00401706.1962.10490022

[15] Rastrigin, L.A. (1974) Systems of Extremal Control. Nauka, Moscow.