A Momentum Trust-Region Algorithm for Unconstrained Optimization

Jose Saavedra
jose.saavedra@cimat.mx

Oscar Dalmau*
dalmau@cimat.mx

Centro de Investigación en Matemáticas
Jalisco S/N, Guanajuato, Gto. 36023, Mexico

August 6, 2026

Abstract

We introduce a Momentum Trust-Region Algorithm for unconstrained optimization that incorporates Nesterov-type acceleration into the classical trust-region framework. The method builds trust-region models around a momentum-shifted point and uses an Armijo-type backtracking procedure to safeguard progress along the resulting displacement. This design preserves the robustness of trust-region methods while exploiting momentum to improve practical efficiency. Under standard smoothness and model-decrease assumptions, we establish global convergence to first-order stationary points. Numerical experiments on 355 functions from the S2MPJ test set show that the proposed method solves more problems and reduces both iteration counts and computational time relative to classical trust-region variants.

*Corresponding author.

1 Introduction

In this work, we consider the unconstrained minimization of a smooth objective function $f : \mathbb{R}^n \to \mathbb{R}$. The problem is to find a point x^* satisfying

$$x^* = \arg\min_{x \in \mathbb{R}^n} \quad f(x) \tag{1}$$

Trust-region methods are classical iterative schemes for solving problems of this form. At each iteration, they replace the objective function by a local model m_k and compute a trial step by approximately minimizing this model inside a region R_k around the current iterate:

$$\begin{aligned} s_k = \arg\min_{s \in \mathbb{R}^n} \quad & m_k(s) \\ \text{s.t.} \quad & s \in R_k \end{aligned} \tag{2}$$

The most common approximation is a quadratic model function $m_k(s) := \frac{1}{2} s^\top B_k s + g_k^\top s + f_k$, where f_k is the value of f at x_k, g_k is the gradient of f at x_k, and B_k is a positive definite matrix. The usual trust region is a Euclidean ball centered at the origin with radius Δ_k. This radius is adjusted at each iteration according to the agreement between the objective function reduction and the model reduction, as measured by the ratio $\tilde{\rho}_k$:

$$\tilde{\rho}_k = \frac{f(x_k) - f(x_k + s_k)}{m_k(0) - m_k(s_k)}. \tag{3}$$

Then, the step s_k is accepted or rejected based on the value of $\tilde{\rho}_k$, and the trust-region radius is increased, decreased, or left unchanged according to prescribed update rules; see, e.g., [14, 18, 5].

In recent years, momentum-based optimization methods have attracted considerable attention due to their strong empirical and theoretical performance across a wide range of applications [15, 13, 11]. By incorporating momentum terms into the iterative process, these methods can promote accelerated convergence and improve the ability of optimization algorithms to navigate complex optimization landscapes, including regions containing saddle points and shallow local minima [19]. Nevertheless, momentum techniques are frequently combined with gradient-descent schemes that employ fixed learning rates, which may limit their efficiency and robustness in practice.

In particular, Nesterov introduced an accelerated gradient method in 1983 [13], often called Nesterov's accelerated gradient. This method has been widely studied and applied in various optimization contexts [2]. The basic scheme is as follows [20]:

$$x_{k+1} = x_k - \alpha \nabla f(x_k + \beta(x_k - x_{k-1})) + \beta(x_k - x_{k-1}). \tag{4}$$

We may introduce an intermediate sequence v_k and split the iteration on three steps:

$$s_k = -\alpha_k \nabla f(x_k + v_k), \tag{5}$$
$$x_{k+1} = x_k + \beta_k v_k + s_k, \tag{6}$$
$$v_{k+1} = v_k + s_k. \tag{7}$$

Here, α_k is called the step size and β_k the momentum parameter. Some recent works have proposed a Nesterov-type acceleration for Quasi-Newton methods, such as BFGS or L-BFGS [12, 16]. In this paper, we propose a Momentum Trust-Region Algorithm that incorporates Nesterov-type acceleration into the trust-region framework.

The main contributions of this work are:

- a momentum trust-region scheme in which the quadratic model is built at a shifted point and the momentum displacement is updated only after an accepted trust-region step;
- a simple, practical mechanism to choose the next displacement using an Armijo-type condition along the momentum direction;
- a global convergence result under standard smoothness and model-decrease assumptions; and
- an empirical evaluation on S2MPJ problems comparing momentum and classical trust-region variants with both exact-Hessian and BFGS model constructions.

The remainder of the paper is organized as follows: Section 2 describes the proposed method and presents the pseudocode, Section 3 establishes global convergence, and Section 4 reports numerical results and performance profiles.

2 The Algorithm

In this work, we propose a Nesterov-type acceleration for trust-region methods applied to problem 1. Starting from a zero initial momentum $v_0 = 0$, zero step length $\beta_0 = 0$, the proposed algorithm iteratively approximates the function by a quadratic model:

$$m_k(s) = \tilde{f}_k + \tilde{g}_k^\top s + \frac{1}{2} s^\top B_k s, \tag{8}$$

where B_k is a symmetric positive definite matrix, $f_k = f(x_k)$, $\tilde{f}_k = f(x_k + \beta_k v_k)$, $g_k = \nabla f(x_k)$ and $\tilde{g}_k = \nabla f(x_k + \beta_k v_k)$. For notational convenience, we will sometimes refer to the intermediate or predicted point:

$$\tilde{x}_{k+1} := x_k + \beta_k v_k. \tag{9}$$

The trust-region sub-problem is then (approximately) solved to obtain a step s_k that minimizes the model within a trust-region of radius Δ_k:

$$\begin{aligned} s_k = \underset{s \in \mathbb{R}^n}{\arg\min} \quad & m_k(s) \\ \text{s.t.} \quad & \|s\|_2 \leq \Delta_k \end{aligned} \tag{10}$$

Then, the ratio $\tilde{\rho}_k$ is computed to evaluate the agreement between the model and the actual function:

$$\tilde{\rho}_k = \frac{f(\tilde{x}_{k+1}) - f(\tilde{x}_{k+1} + s_k)}{m_k(0) - m_k(s_k)}. \tag{11}$$

Based on the value of $\tilde{\rho}_k$, the step s_k is accepted or rejected, giving rise to the next iterate x_{k+1}. Given constants $0 < r_0 \leq r_1 < \frac{1}{2} \leq r_2 < 1$, if $\tilde{\rho}_k > r_0$, the step is accepted and the iterates are updated as follows:

$$x_{k+1} = \tilde{x}_{k+1} + s_k = x_k + \beta_k v_k + s_k, \tag{12}$$

$$g_{k+1} = \nabla f(x_{k+1}), \tag{13}$$

$$v_{k+1} = \beta_k v_k + s_k, \tag{14}$$

and β_{k+1} is chosen in $\left(0, \frac{\Delta_k}{\|v_k\|}\right)$ to satisfy the Armijo [1] condition:

$$f(x_{k+1} + \beta_{k+1} v_{k+1}) \leq f_{k+1} + w \beta_{k+1} g_{k+1}^\top v_{k+1}. \tag{15}$$

In case the step is rejected, the iterates are updated as follows:

$$x_{k+1} = x_k, \tag{16}$$
$$g_{k+1} = g_k, \tag{17}$$
$$v_{k+1} = v_k, \tag{18}$$
$$\beta_{k+1} = \beta_k. \tag{19}$$

Algorithm 1 Momentum Trust-Region Algorithm.

Require: $f, x_0, \eta > 1, 0 < w < 1, 0 < r_0 \leq r_1 < 1/2 \leq r_2 < 1, 0 < \Delta_- < \Delta_0 < \Delta_+, \epsilon > 0$
Ensure: x^*

$k \leftarrow 0$
$v_0 \leftarrow \mathbf{0}$
$\beta_0 \leftarrow 0$
while $\|g_k\|_2 > \epsilon$ **do**
  $\tilde{x}_{k+1} \leftarrow x_k + \beta_k v_k$
  Find s_k an exact or approximate solution to the sub-problem (10)
  $\tilde{\rho}_k \leftarrow \frac{f(\tilde{x}_{k+1}) - f(\tilde{x}_{k+1} + s_k)}{m_k(0) - m_k(s_k)}$
  if $\tilde{\rho}_k > r_0$ **then**
    $x_{k+1} \leftarrow \tilde{x}_{k+1} + s_k$
    Find $\beta_{k+1} \in (0, \Delta_k / \|v_k\|)$ such that $f(x_{k+1} + \beta_{k+1} v_{k+1}) \leq f_{k+1} + w\beta_{k+1} g_{k+1}^\top v_{k+1}$
  else
    $x_{k+1} \leftarrow x_k$
    $v_{k+1} \leftarrow v_k$
    $\beta_{k+1} \leftarrow \beta_k$
  end if
  if $\tilde{\rho}_k < r_1$ **then**
    $\Delta_{k+1} \leftarrow \frac{1}{\eta} \Delta_k$
  else if $\tilde{\rho}_k > r_2$ **then**
    $\Delta_{k+1} \leftarrow \eta \Delta_k$
  else
    $\Delta_{k+1} \leftarrow \Delta_k$
  end if
  $k \leftarrow k + 1$
end while

At the end of each iteration, we update the radius of the trust-region Δ_{k+1} as follows:

$$\Delta_{k+1} = \begin{cases} \frac{1}{\eta}\Delta_k, & \text{if } \tilde{\rho}_k < r_1, \\ \Delta_k, & \text{if } r_1 \leq \tilde{\rho}_k \leq r_2, \\ \eta\Delta_k, & \text{if } \tilde{\rho}_k > r_2, \end{cases} \tag{20}$$

for some constant $\eta > 1$. The complete pseudocode is presented in Algorithm 1.

3 Convergence analysis

The convergence analysis of the proposed algorithm is very similar to that of the classical trust-region method, such as the one by Shultz, Schnabel and Byrd [18]. First, we make the following assumptions:

Assumption 1: The objective function is Lipschitz continuously differentiable, i.e.

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq \nu_1 \|x - y\|_2 \tag{21}$$

for some $\nu_1 > 0$.

Assumption 2: There is a constant $\nu_2 > 0$ such that each matrix B_k satisfies

$$\|B_k\|_2 \leq \nu_2, \quad \forall k \tag{22}$$

Assumption 3: The function reduction obtained by step s_k satisfies

$$m_k(0) - m_k(s_k) \geq c_1 \|\tilde{g}_k\|_2 \min\left\{\Delta_k, \frac{\|\tilde{g}_k\|_2}{\|B_k\|_2}\right\} \tag{23}$$

for some $c_1 \in (0, 1)$.

Assumption 4: The step s_k is inside the trust-region, i.e. $\|s_k\|_2 \leq \Delta_k$.

Assumption 5: The objective function is bounded below on the level set $\mathcal{L} := \{x \mid f(x) \leq f(x_0)\}$.

Assumption 6: The momentum step is always a descent direction and the cosine of the angle between the momentum step and the negative gradient is bounded below, i.e. there exists a constant $c_2 \in (0,1)$ such that $-g_k^\top v_k \geq \|g_k\|\|v_k\|c_2 > 0$ for all $k > 0$.

Theorem 1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function satisfying Assumption 1–Assumption 6. Then, the sequence $\{g_k\}$ generated by the Algorithm 1 satisfies $\lim_{k\to\infty} \|g_k\| = 0$.*

Proof. Begin by considering an index m such that $\|g_m\| \neq 0$. Then, for any x,

$$\|\nabla f(x) - g_m\| \leq \nu_1 \|x - x_m\|.$$

Let $\epsilon = \frac{\|g_m\|}{2}$, $R = \frac{\epsilon}{\nu_1}$ and $B_R = \{x : \|x - x_m\| \leq R\}$. The sequence generated by the algorithm has two possibilities: either x_k and $\tilde{x}_k \in B_R$ for all $k \geq m$, or there is an index $K > m$ such that either $x_K \notin B_R$ or $\tilde{x}_K \notin B_R$. We shall prove that the first case is impossible.
Assume by means of contradiction that x_k and $\tilde{x}_k \in B_R$ for all $k \geq m$. Then,

$$|\tilde{\rho}_k - 1| = \left| \frac{m_k(s_k) - f(\tilde{x}_k + s_k)}{m_k(0) - m_k(s_k)} \right|$$

Now, we expand $f(\tilde{x}_k + s_k)$:

$$f(\tilde{x}_k + s_k) = f(\tilde{x}_k) + \tilde{g}_k^\top s_k + \int_0^1 \left[g(\tilde{x}_k + t s_k) - \tilde{g}_k \right]^\top s_k dt, \tag{24}$$

and we may bound $|m_k(s_k) - f(\tilde{x}_k + s_k)|$ as follows:

$$\begin{aligned} |m_k(s_k) - f(\tilde{x}_k + s_k)| &= \left| \frac{1}{2} s_k^\top B_k s_k - \int_0^1 \left[g(\tilde{x}_k + t s_k) - \tilde{g}_k \right]^\top s_k dt \right|, \\ &\leq \frac{\nu_2 + \nu_1}{2} \|s_k\|^2. \end{aligned} \tag{25}$$

Now, for any $x \in B_R$, we have:

$$\|\nabla f(x)\| \geq \|g_m\| - \|\nabla f(x) - g_m\| \geq \frac{1}{2}\|g_m\| = \epsilon. \tag{26}$$

Then, $\|g_k\| \geq \epsilon$ and $\|\tilde{g}_k\| \geq \epsilon$ for all $k \geq m$. From Assumption 3, we have that:

$$\begin{aligned} m_k(0) - m_k(s_k) &\geq c_1 \|\tilde{g}_k\| \min\left\{\Delta_k, \frac{\|\tilde{g}_k\|}{\|B_k\|}\right\}, \\ &\geq c_1 \epsilon \min\left\{\Delta_k, \frac{\epsilon}{\nu_2}\right\}. \end{aligned} \tag{27}$$

Then, from (25) and (27) we have:

$$|\tilde{\rho}_k - 1| \leq \frac{\Delta_k^2(\nu_1 + \nu_2)}{2c_1\epsilon \min\{\Delta_k, \epsilon/\nu_2\}}. \tag{28}$$

Now, we define

$$\hat{\Delta} = \frac{c_1\epsilon}{\nu_1 + \nu_2}. \tag{29}$$

Note $\hat{\Delta} \leq \epsilon/\nu_2$ since $c_1 \leq 1$ and $\nu_1, \nu_2 > 0$. Now, from the previous inequality, we have that if $\Delta_k \leq \hat{\Delta}$, then

$$|\tilde{\rho}_k - 1| \leq \frac{\Delta_k}{2\hat{\Delta}} \leq 1/2.$$

Which implies that $\tilde{\rho}_k \geq 1/2 > r_1$ and the radius Δ_k is never reduced below Δ_k whenever $\Delta_k \leq \hat{\Delta}$. Combining both cases, a decrease of the trust-region radius Δ_k can occur only if $\Delta_k > \hat{\Delta}$, obtaining a lower bound on the radius Δ_k:

$$\Delta_k \geq \min\left\{\Delta_m, \hat{\Delta}/\eta\right\} > 0. \tag{30}$$

Next, suppose that there is an infinite sub-sequence of indices $\mathcal{K}$ such that $\tilde{\rho}_k \geq r_1$ for all $k \in \mathcal{K}$. Then, for all $k \in \mathcal{K}, k \geq m$, we have

$$\begin{aligned} f_k - f_{k+1} &= f(x_k) - f(\tilde{x}_{k+1} + s_k), \\ &\geq r_1 \left(m_k(0) - m_k(s_k)\right), \end{aligned}$$

and from (27)

$$f_k - f_{k+1} \geq r_1 c_1 \epsilon \min\left\{\Delta_k, \frac{\epsilon}{\nu_2}\right\} \tag{31}$$

Since k is bounded below, we have that

$$\lim_{k \in \mathcal{K}, k \to \infty} \Delta_k = 0$$

contradicting (30), the lower bound on Δ_k. Then, no such infinite subsequence $\mathcal{K}$ may exist, and there is an index $K \geq m$ such that $\tilde{\rho}_k < r_1$ for all $k \geq K$. Then, for all $k \geq K$, we have that $\Delta_{k+1} = \Delta_k/\eta$, yielding

$$\lim_{k\to\infty} \Delta_k = 0,$$

which again contradicts (30). Therefore, the original assumption that x_k and $\tilde{x}_k \in B_R$ for all $k \geq m$ is impossible, and there exists an index $l > m$ such that either $x_l \notin B_R$ or $\tilde{x}_l \notin B_R$.

Let l be the first index such that x_l or $\tilde{x}_l \notin B_R$.

$$\begin{aligned}
f_m - f_l &= \sum_{k=m}^{l-1} f_k - f_{k+1}, \\
&= \sum_{\substack{k=m \\ x_{k+1}\neq x_k}}^{l-1} f_k - f(x_k + \beta_k v_k + s_k), \\
&= \sum_{\substack{k=m \\ x_{k+1}\neq x_k}}^{l-1} f_k - \tilde{f}_k + \tilde{f}_k - f(x_k + \beta_k v_k + s_k), \\
&\geq \sum_{\substack{k=m \\ x_{k+1}\neq x_k}}^{l-1} w\beta_k(-g_k^\top v_k) + r_0\left(m_k(0) - m_k(s_k)\right), \\
&\geq \sum_{\substack{k=m \\ x_{k+1}\neq x_k}}^{l-1} w\beta_k(-g_k^\top v_k) + r_0 c_1 \|\tilde{g}_k\| \min\left\{\Delta_k, \frac{\|\tilde{g}_k\|}{\nu_2}\right\},
\end{aligned}$$

From (26) and Assumption 6, we have $\|\tilde{g}_k\| \geq 1/2\|g_m\|$ and

$$\begin{aligned}
f_m - f_l &\geq \sum_{\substack{k=m \\ x_{k+1}\neq x_k}}^{l-1} w\beta_k(-g_k^\top v_k) + r_0 c_1 \frac{1}{2}\|g_m\| \min\left\{\Delta_k, \frac{\|g_m\|}{2\nu_2}\right\}, \\
&\geq \sum_{\substack{k=m \\ x_{k+1}\neq x_k}}^{l-1} w\|\beta_k v_k\|\|g_k\| c_2 + r_0 c_1 \frac{1}{2}\|g_m\| \min\left\{\Delta_k, \frac{\|g_m\|}{2\nu_2}\right\}, \\
&\geq \sum_{\substack{k=m \\ x_{k+1}\neq x_k}}^{l-1} \frac{1}{2} w c_2 \|\beta_k v_k\|\|g_m\| + r_0 c_1 \frac{1}{2}\|g_m\| \min\left\{\Delta_k, \frac{\|g_m\|}{2\nu_2}\right\}. \qquad (32)
\end{aligned}$$

Define

$$u = \min\{wc_2, r_0c_1\} > 0, \tag{33}$$

and we obtain a lower bound for the right-hand side of inequality (32):

$$f_m - f_l \geq \sum_{\substack{k=m \\ x_{k+1}\neq x_k}}^{l-1} \frac{1}{2}u\|g_m\| \left(\|\beta_k v_k\| + \min\left\{\Delta_k, \frac{\|g_m\|}{2\nu_2}\right\}\right). \tag{34}$$

In case $\Delta_k \leq \frac{\|g_m\|}{2\nu_2}$ for all $m \leq k < l$, then

$$\begin{aligned} f_m - f_l &\geq \sum_{\substack{k=m \\ x_{k+1}\neq x_k}}^{l-1} \frac{1}{2}u\|g_m\| \left(\|\beta_k v_k\| + \Delta_k\right), \\ f_m - f_l &\geq \frac{1}{2}u\|g_m\|R. \end{aligned} \tag{35}$$

Otherwise, there is an index $K \leq k < l$ such that $\Delta_k > \frac{\|g_m\|}{2\nu_2}$. Then,

$$\begin{aligned} f_m - f_l &\geq \frac{1}{2}u\|g_m\|\frac{\|g_m\|}{2\nu_2} + \sum_{\substack{k=m \\ x_{k+1}\neq x_k}}^{l-1} \frac{1}{2}u\|g_m\|\|\beta_k v_k\| \\ &\geq u\frac{1}{2}\|g_m\|\frac{\|g_m\|}{2\nu_2}. \end{aligned} \tag{36}$$

In both cases, we have that

$$\begin{aligned} f_m - f_l &\geq u\frac{1}{2}\|g_m\| \min\left\{\frac{\|g_m\|}{2\nu_2}, R\right\}, \\ &\geq u\frac{1}{4}\ \|g_m\|^2 \min\left\{\frac{1}{\nu_1}, \frac{1}{\nu_2}\right\}. \end{aligned} \tag{37}$$

Since the sequence f_k is bounded below and monotonically decreasing, there exists $f^* > -\infty$ such that $f^* = \lim_{k\to\infty} f_k$, then

$$f(x_m) - f^* \geq f_m - f_l, \tag{38}$$

and since

$$u\frac{1}{4}\|g_m\|^2 \min\left\{\frac{1}{\nu_1}, \frac{1}{\nu_2}\right\} > 0, \tag{39}$$

taking the limit as $m \to \infty$ for the right-hand side of inequality (37), we obtain

$$\lim_{m\to\infty} \|g_m\| = 0. \tag{40}$$

□

4 Numerical experiments

As a numerical challenge for Algorithm 1, we compare its performance against the classical trust-region method on problems from the S2MPJ test set [10] using Julia [3]. S2MPJ is a MATLAB, Python and Julia interface to a large collection of optimization test problems, including standard unconstrained problems commonly used to benchmark nonlinear optimization algorithms. It provides problem definitions, derivatives, and standard initial points in a unified format, making it suitable for reproducible comparisons across solvers. We evaluate the methods on 355 unconstrained problems, corresponding to the available unconstrained S2MPJ instances after excluding a small number of cases that could not be run reliably because of external-library issues, such as LAPACK failures.

As the approximate sub-problem solution, we use the Dogleg method[6, 14] due to its efficiency and robustness. We report results for five solvers:

1. Classical trust-region with exact Hessian:

$$B_k = \nabla^2 f(x_k), \tag{41}$$

2. Classical trust-region with a BFGS [4, 8, 9, 17] approximation:

$$y_k = g_{k+1} - g_k, \tag{42}$$

$$B_{k+1} = B_k - \frac{B_k s_k s_k^\top B_k}{s_k^\top B_k s_k} + \frac{y_k y_k^\top}{y_k^\top s_k}, \tag{43}$$

where s_k is the accepted step and $g_k = \nabla f(x_k)$.

3. Classical trust-region with a damped BFGS update [14]:

$$y_k = g_{k+1} - g_k, \tag{44}$$
$$\tilde{y}_k = \theta_k y_k + (1 - \theta_k) B_k s_k, \tag{45}$$
$$B_{k+1} = B_k - \frac{B_k s_k s_k^\top B_k}{s_k^\top B_k s_k} + \frac{\tilde{y}_k \tilde{y}_k^\top}{\tilde{y}_k^\top s_k}, \tag{46}$$

where s_k is the accepted step, $g_k = \nabla f(x_k)$, and the damping parameter θ_k is defined as

$$\theta_k = \begin{cases} 1, & \text{if } y_k^\top s_k \geq 0.2\, s_k^\top B_k s_k, \\ 0.8 \dfrac{s_k^\top B_k s_k}{s_k^\top B_k s_k - s_k^\top y_k}, & \text{if } y_k^\top s_k < 0.2\, s_k^\top B_k s_k. \end{cases} \tag{47}$$

This damping mechanism ensures the curvature condition $\tilde{y}_k^\top s_k > 0$ holds even when $y_k^\top s_k \leq 0$, which can occur in non-convex problems.

4. Momentum Trust-Region with exact Hessian evaluated at the shifted point:

$$B_k = \nabla^2 f(\tilde{x}_k), \tag{48}$$

5. Momentum Trust-Region with a damped shifted BFGS approximation:

$$y_k = g_{k+1} - g_k, \tag{49}$$
$$\tilde{y}_k = \theta_k y_k + (1 - \theta_k) B_k v_{k+1}, \tag{50}$$
$$B_{k+1} = B_k + \frac{\tilde{y}_k \tilde{y}_k^\top}{\tilde{y}_k^\top v_{k+1}} - \frac{B_k v_{k+1} v_{k+1}^\top B_k}{v_{k+1}^\top B_k v_{k+1}}, \tag{51}$$

where v_{k+1} is given by 14 and the damping parameter θ_k is defined as in solver 41, ensuring $\tilde{y}_k^\top v_{k+1} > 0$.

6. Momentum Trust-Region with a shifted BFGS approximation:

$$y_k = g_{k+1} - g_k, \tag{52}$$
$$B_{k+1} = B_k + \frac{y_k y_k^\top}{y_k^\top v_{k+1}} - \frac{B_k v_{k+1} v_{k+1}^\top B_k}{v_{k+1}^\top B_k v_{k+1}}, \tag{53}$$

where v_{k+1} is given by 14.

Throughout Tables 1 and 2, as well as the performance-profile figures 1 and 2, CH denotes classical trust-region with exact Hessian, CB denotes classical trust-region with a BFGS update, CD denotes classical trust-region with a damped BFGS update, and MH, MB, and MD denote the corresponding momentum trust-region variants.

Experimental setup. All methods start from the standard S2MPJ initial point x_0 and stop when $\|\nabla f(x_k)\|_2 \leq 10^{-4}$, when the iteration counter reaches $k = 10^5$, or when a wall-clock timeout of one hour is reached for a given problem. The trust-region parameters are fixed across all runs as follows: acceptance threshold $r_0 = 0.1$, radius-update thresholds $r_1 = \frac{1}{4}$ and $r_2 = \frac{1}{2}$, update factor $\eta = 2$, and radius bounds $\Delta_{\min} = 0.01\sqrt{n}$ and $\Delta_{\max} = 10\sqrt{n}$. The Dogleg method is used as the approximate solver of the trust-region sub-problem.

In practice, it is unrealistic to expect a fixed momentum construction to satisfy the descent-direction condition $g_k^\top v_k < 0$ for arbitrary non-convex objectives. Therefore, the implementation includes a simple safeguard: if $g_k^\top v_k > 0$, we flip the sign of the momentum displacement by replacing $v_k \leftarrow -v_k$. This ensures the method never uses an ascent momentum displacement.

Computing environment. All experiments were executed on a workstation with an AMD Ryzen 9 5900X (12 cores, 4.20 GHz), 96 GB RAM, running Windows 11 (25H2), using Julia 1.12.5 with `JULIA_NUM_THREADS=8`. Julia multi-threading is used only to parallelize over the problem list (each thread solves a different test problem). For any fixed problem and algorithm, the solver itself runs on a single thread, so the reported time is the per-solve wall-clock time of one run.

Initialization and line searches. The initial radius is set to $\Delta_0 = 1$ and then decreased by backtracking along the steepest-descent direction $-g_0$ until the Armijo condition

$$f(x_0 - \Delta_0 g_0) \leq f_0 - w\,\Delta_0\, g_0^\top g_0, \tag{54}$$

holds, with Armijo parameter $w = 10^{-4}$, contraction factor $\lambda = 0.8$, and at most 30 backtracking steps (fallback $\Delta_0 = \|\nabla f(x_0)\|_2$ if function evaluations fail). For the momentum method, the Armijo search is applied to the

momentum displacement by scaling $v_{k+1} \leftarrow \lambda v_{k+1}$ until

$$f(x_{k+1} + v_{k+1}) \leq f(x_{k+1}) + w\, \nabla f(x_{k+1})^\top v_{k+1} \tag{55}$$

holds (again with $w = 10^{-4}$, $\lambda = 0.8$, and at most 30 steps). In the implementation, this is equivalent to absorbing the scalar β_{k+1} into the stored vector v_{k+1}.

Hessian and BFGS safeguards. When using exact second derivatives, the Hessian is evaluated at x_k for the classical trust-region method and at the shifted point $x_k + v_k$ for the momentum method. In both cases, if the Hessian is ill-conditioned or indefinite, it is regularized to be symmetric positive definite using a Bunch-Kaufman factorization and a diagonal modification. Both the classical and momentum trust-region variants employ the same damped BFGS update [14] to ensure curvature conditions are satisfied. As noted in Section 4, the damped variants underperform compared to their standard BFGS counterparts in this non-convex test set.

Performance profiles. Performance profiles are constructed following the standard definition in [7] using both iteration counts and wall-clock times (measured in Julia with `@elapsed`).

The iteration and CPU time performance profiles are presented in Figures 1 and 2, respectively. The iteration profile shows that the proposed method outperforms the classical algorithm, specially with the exact Hessian, as expected, while the CPU time profile shows that the classical algorithm with the BFGS update is the fastest but solves the fewest problems, while the proposed algorithm is marginally slower but outperforms it very soon as it solves more problems, as well as outperforming the classical algorithm with the exact Hessian.

It is also interesting to note that the exact-Hessian implementation of Algorithm 1 is faster than its BFGS counterpart, suggesting the iteration advantage is large enough to forego Hessian approximation and still win. Overall, the proposed algorithm offers better performance than the classical trust-region method in terms of problems solved, iterations, and CPU time.

Discussion. Taken together, these profiles indicate that the momentum mechanism can deliver a substantial reduction in iteration counts, and that this reduction can translate into improved time-to-solution across a nontrivial

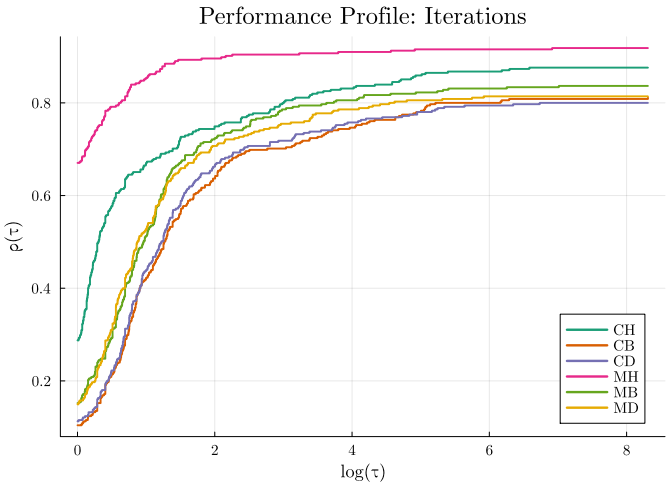


Figure 1: Performance profiles of both algorithms using exact Hessian and a BFGS approximation. Iteration profile.

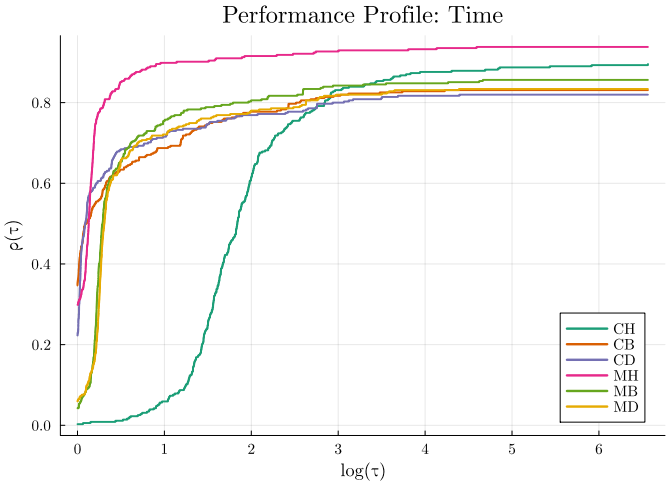


Figure 2: Performance profiles of both algorithms using exact Hessian and a BFGS approximation. CPU time profile.

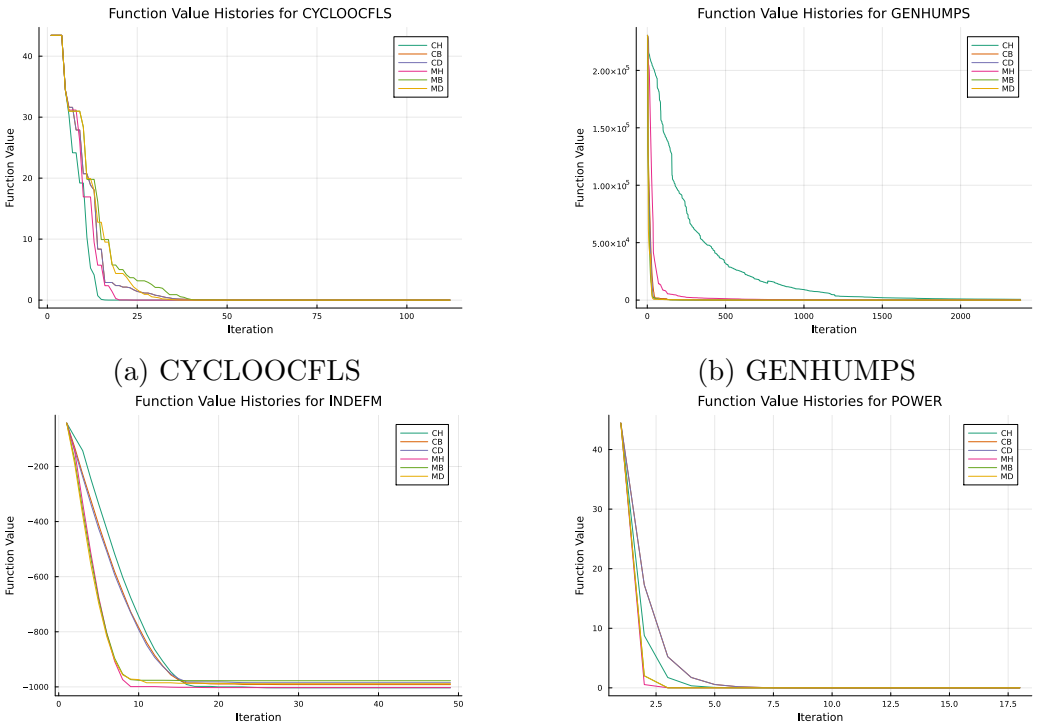
subset of the test set. The time profile also highlights a common trade-off: a method with cheaper iterations (here, classical trust-region with BFGS) can appear favorable early in the profile while still failing to solve as many problems within the chosen stopping criteria. In contrast, the momentum method can be slightly more expensive per iteration yet become preferable as soon as robustness (problems solved) is taken into account, being the safest option in terms of overall performance.

Practical takeaways from the experiments. Across the tested problems, the momentum variant improves the iteration profile and, in many cases, improves time-to-solution relative to the classical trust-region baseline. The profiles also suggest that the momentum mechanism can make the exact-Hessian variant competitive (or even preferable) despite its higher per-iteration algebraic cost, indicating that the iteration reduction can be large enough to offset second-derivative overhead in this test setting.

Case studies of representative functions. To gain further insight into the behavior observed in the performance profiles, we selected several representative functions where the momentum method consistently demonstrates its advantages. The functions CYCLOOCFLS, GENHUMPS, INDEFM, and POWER exhibit a pattern where the momentum method is not always the absolute fastest solver, but it consistently ranks among the top performers, offering a balanced combination of speed and robustness. Figure 3 shows the function value history for these functions, plotting $f(x_k)$ as a function of iteration count. In each case, the momentum variants (especially the damped BFGS variant) achieve faster reduction in the function value, demonstrating the practical benefits of the Nesterov acceleration within the trust-region framework.

Figure 4 shows the gradient norm history for the same functions, plotting $\|\nabla f(x_k)\|$ as a function of iteration count. Again, the momentum variants demonstrate faster reduction in the gradient norm, confirming their effectiveness in driving the iterates toward first-order stationality.

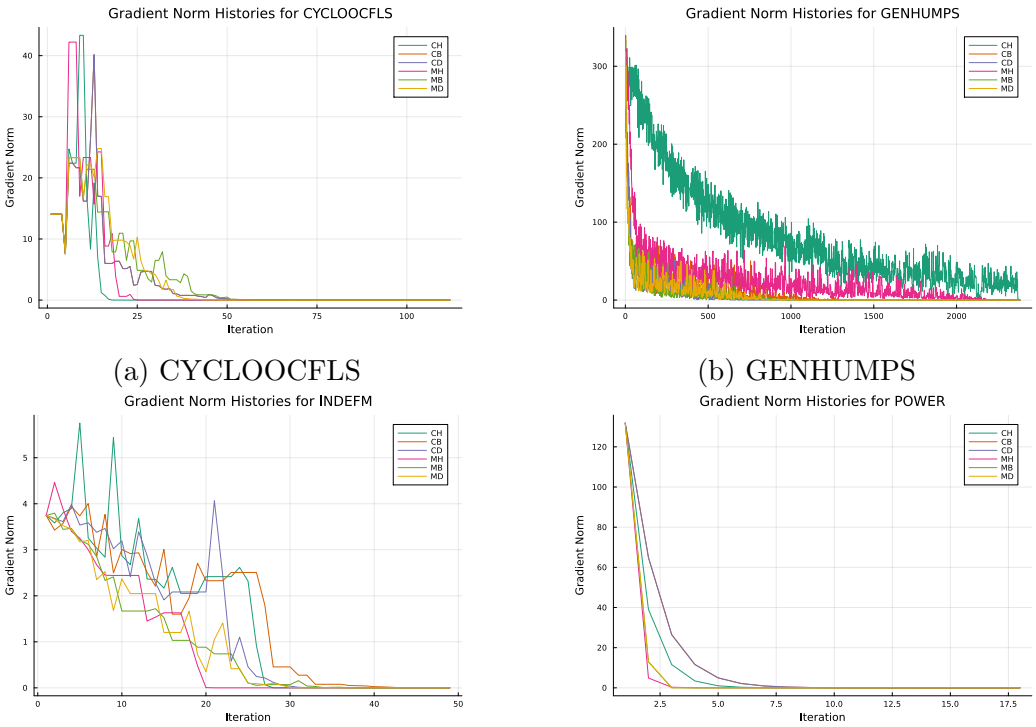
Dramatic improvements on challenging problems. In addition to the case studies above, we identified several challenging functions where the classical trust-region method converges very slowly, while the momentum variants achieve substantial acceleration. For these problems, the momentum



(a) CYCLOOCFLS (b) GENHUMPS

(c) INDEFM (d) POWER

Figure 3: Function value history for representative functions.



(a) CYCLOOCFLS

(b) GENHUMPS

(c) INDEFM

(d) POWER

Figure 4: Gradient norm history for representative functions.

mechanism appears to be particularly effective at overcoming barriers that impede the classical method's progress. The tables below report exactly this subset: the four case-study functions (CYCLOOCFLS, GENHUMPS, INDEFM, and POWER) together with the additional challenging problems identified above. While detailed plots for these cases are omitted for brevity, their performance is included in Table 1, which compares the total number of iterations, as well as in Table 2, which compares wall-clock time for the classical and momentum variants. Both tables also include the median iteration count and CPU time across all problems for each solver.

Function	CH	CB	CD	MH	MB	MD
BIGGS3	**71**	95	120	2972	**68**	78
CURLY10	20	31	**26**	**10**	35	23
CYCLOOCFLS	**31**	112	112	**36**	62	61
DJTL	**555**	298	498	1045	114	**102**
FMINSRF2	115	38	**38**	**28**	49	49
GENHUMPS	2383	1353	**788**	1751	1499	944
GULF	12078	15495	17282	143	150	**124**
HIMMELBF	18459	18477	21277	**115**	**166**	2599
INDEFM	28	**49**	39	**17**	32	24
PFIT4LS	59315	**269**	1205	345	552	557
POWER	13	18	18	8	**5**	9
SINQUAD2	103	127	48	**8**	41	21
SPARSQUR	11	16	16	**7**	**4**	7
VESUVIOULS	17620	18124	20616	3079	**1858**	2670
Median	**109**	**120**	**116**	**76**	**65**	**70**

Table 1: Iteration counts for the case-study functions and additional challenging problems from the S2MPJ test set. The best performance for each problem is highlighted in bold.

Limitations. There are several aspects that warrant care in interpreting and extending these results. First, the convergence analysis assumes that whenever the momentum displacement is used it is a descent direction; for general non-convex objectives this can fail without safeguards, motivating the practical sign-flip mechanism described in the implementation. Second, the method introduces additional algorithmic parameters (e.g., Armijo constants

Function	CH	CB	CD	MH	MB	MD
BIGGS3	**6.05**	6.79	6.87	7.20	**6.80**	6.80
CURLY10	1.29	1.29	**1.29**	**1.29**	1.29	1.29
CYCLOOCFLS	**4.13**	4.14	4.14	**4.14**	4.14	4.14
DJTL	**5.59**	6.17	6.18	5.60	6.17	**6.17**
FMINSRF2	1.69	1.68	**1.68**	**1.68**	1.68	1.68
GENHUMPS	1.96	1.67	**1.59**	1.88	1.81	1.71
GULF	13.50	7.27	7.86	1.90	1.86	**1.82**
HIMMELBF	3.87	2.90	2.87	**2.10**	**2.09**	2.28
INDEFM	1.38	**1.38**	1.38	**1.38**	1.38	1.38
PFIT4LS	5.66	**1.95**	1.98	1.98	1.98	1.98
POWER	1.31	1.31	1.31	1.31	**1.31**	1.31
SINQUAD2	1.59	1.59	1.58	**1.58**	1.59	1.58
SPARSQUR	1.60	1.60	1.60	**1.60**	**1.60**	1.60
VESUVIOULS	365.70	173.93	190.51	101.03	**55.81**	74.42
Median	**2.92**	**1.81**	**1.83**	**1.89**	**1.84**	**1.77**

Table 2: Wall-clock time (in seconds) for the case-study functions and additional challenging problems from the S2MPJ test set. The best performance for each problem is highlighted in bold.

and backtracking limits) whose tuning can influence performance across problem classes. Finally, while the global convergence result establishes stationarity, this work does not attempt to derive iteration-complexity bounds or local convergence rates.

5 Conclusion

Nesterov-type acceleration has been shown to substantially improve the practical performance of first-order and quasi-Newton methods, but its interaction with trust-region mechanisms has received comparatively little attention. This work proposed a *Momentum Trust-Region Algorithm* that augments a standard trust-region framework with a momentum displacement and an Armijo-type backtracking procedure along that displacement. The resulting method retains the classical accept/reject logic of trust-region schemes while using momentum to bias subsequent models toward directions that empirically improve progress on challenging non-convex landscapes.

Future work. Several directions appear natural. On the theory side, it would be useful to (i) relax or replace the descent-direction assumption on the momentum displacement, (ii) establish evaluation-complexity bounds analogous to those available for classical trust-region methods, and (iii) study local behavior near minimizers (including the interplay with quasi-Newton updates at shifted points). On the algorithmic side, promising extensions include adaptive strategies for choosing the displacement and backtracking parameters, alternative sub-problem solvers and preconditioning for large-scale instances, and variants tailored to stochastic or noisy objectives.

Acknowledgements

Jose Saavedra gratefully acknowledges financial support from the Secretaría de Ciencia, Humanidades, Tecnología e Innovación (SECIHTI), Mexico, through a graduate scholarship.

References

[1] L. Armijo, *Minimization of functions having lipschitz continuous first partial derivatives*, Pacific Journal of Mathematics, 16 (1966), p. 1–3.

[2] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM Journal on Imaging Sciences, 2 (2009), p. 183–202.

[3] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, *Julia: A fresh approach to numerical computing*, SIAM Review, 59 (2017), pp. 65–98.

[4] C. G. Broyden, *The convergence of a class of double-rank minimization algorithms 1. general considerations*, IMA Journal of Applied Mathematics, 6 (1970), p. 76–90.

[5] F. E. Curtis, Z. Lubberts, and D. P. Robinson, *Concise complexity analyses for trust region methods*, Optimization Letters, 12 (2018), p. 1713–1724.

[6] P. M. J. D., *A hybrid method for nonlinear equations*, Numerical Methods for Nonlinear Algebraic Equations, (1970), pp. 87–161.

[7] E. D. Dolan and J. J. Moré, *Benchmarking optimization software with performance profiles*, Mathematical Programming, 91 (2002), p. 201–213.

[8] R. Fletcher, *A new approach to variable metric algorithms*, The Computer Journal, 13 (1970), p. 317–322.

[9] D. Goldfarb, *A family of variable-metric methods derived by variational means*, Mathematics of Computation, 24 (1970), p. 23–26.

[10] S. Gratton and P. L. Toint, *S2MPJ and CUTEst optimization problems for Matlab, Python and Julia*, Optimization Methods and Software, 40 (2025), p. 871–903.

[11] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, in 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, Y. Bengio and Y. LeCun, eds., 2015.

[12] S. Mahboubi, I. S, H. Ninomiya, and H. Asai, *Momentum acceleration of quasi-Newton based optimization technique for neural network training*, Nonlinear Theory and Its Applications, IEICE, 12 (2021), p. 554–574.

[13] Y. E. Nesterov, *A method of solving a convex programming problem with convergence rate $O(1/k^2)$*, Soviet Mathematics Doklady, 27 (1983), pp. 372–376. English translation of Dokl. Akad. Nauk SSSR **269** (1983), 543–547.

[14] J. Nocedal and S. Wright, *Numerical Optimization*, Springer New York, New York, 2 ed., 2006.

[15] B. Polyak, *Some methods of speeding up the convergence of iteration methods*, USSR Computational Mathematics and Mathematical Physics, 4 (1964), p. 1–17.

[16] M. K. Sahu and S. R. Pattanaik, *Non-asymptotic superlinear convergence of Nesterov accelerated BFGS*, Research Square, (2026). Preprint.

[17] D. F. Shanno, *Conditioning of quasi-newton methods for function minimization*, Mathematics of Computation, 24 (1970), p. 647–656.

[18] G. A. Shultz, R. B. Schnabel, and R. H. Byrd, *A Family of Trust-Region-Based Algorithms for Unconstrained Minimization with Strong Global Convergence Properties*, SIAM Journal on Numerical Analysis, 22 (1985), p. 47–67.

[19] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, *On the importance of initialization and momentum in deep learning*, in Proceedings of the 30th International Conference on Machine Learning, S. Dasgupta and D. McAllester, eds., vol. 28 of Proceedings of Machine Learning Research, Atlanta, Georgia, USA, 17–19 Jun 2013, PMLR, pp. 1139–1147.

[20] S. J. Wright and B. Recht, *Optimization for Data Analysis*, Cambridge University Press, New York, 2022.