# Stochastic Three Points Method with an Inexact Oracle and Its Application to Steady-State Optimization

Geovani Nunes Grapiglia*

April 23, 2026

**Abstract**

We consider unconstrained derivative-free optimization problems in which only inexact function evaluations are available. Specifically, we study the setting where the oracle returns function values with partially controllable inexactness, with the error bounded linearly by a user-specified accuracy parameter, but with an unknown proportionality constant. This framework captures optimization problems arising from approximate simulations or experimental evaluations with adjustable accuracy. We propose a variant of the Stochastic Three Points method (SIAM J. Optim. 30, 1–25, 2020) that jointly updates oracle accuracy and stepsizes without requiring prior knowledge of the error bound. Assuming that the objective function has a Lipschitz continuous gradient, we show that the proposed method requires at most $\mathcal{O}\big(n|\ln(\epsilon)|^2\epsilon^{-2}\big)$ calls to the inexact oracle to find a point whose expected gradient norm is below $\epsilon$, where $n$ is the number of variables. As an application, we study model-free steady-state optimization in control systems and show that it can be addressed using the proposed method.

## 1 Introduction

Derivative-free optimization (DFO) concerns the solution of optimization problems in which gradient information of the objective function is unavailable [10, 3, 26], forcing the optimization solvers to rely only on function evaluations. DFO problems appear in a wide range of applications, including hyperparameter tuning in machine learning [32], engineering optimization [16, 29], design of black-box adversarial attacks [9], bilevel optimization [15, 8], and control of physical systems [25, 21, 27]. Several oracle models have been studied in the DFO literature. In the simplest case, the oracle returns exact function values. However, in many applications, one often only has access to inexact oracles, which provide function values corrupted by stochastic or deterministic errors arising from finite-precision computations or approximate simulations. Depending on the oracle type and the problem dimension, different classes of methods have been developed. For low- to moderate-dimensional problems, model-based trust-region [30, 11, 18, 36], direct search [34, 22, 12, 5, 13] and finite-difference methods [1, 19, 31, 23] are widely used. In higher-dimensional settings, randomized methods [28, 20, 9, 2, 7, 14] are often preferred due to their scalability.

In this paper, we consider a class of DFO problems characterized by an oracle with *partially controllable inexactness*. Specifically, for any query point $u \in \mathbb{R}^n$ and any accuracy parameter

*Université catholique de Louvain, 1348 Louvain-la-Neuve, Belgium (geovani.grapiglia@uclouvain.be). This author was partially supported by FRS-FNRS, Belgium (Grant CDR J.0094.26).

$\delta > 0$, the oracle returns a value $f_\delta(u)$ satisfying $|f_\delta(u) - f(u)| \leq C\delta$, where $C > 0$ is an unknown constant that is independent of $u$ and $\delta$. This oracle model represents scenarios where accuracy can be improved at the cost of more computation or waiting time, though the exact error is unknown. Such oracles fall outside the scope of classical DFO frameworks and motivate the development of new algorithms and analyses. Specifically, we propose a variant of the Stochastic Three Points (STP) method [2] tailored to this type of inexact oracle. The method adaptively balances the oracle accuracy parameter $\delta$ with the optimization progress, without requiring knowledge of the constant $C$. We show that the proposed method requires at most $\mathcal{O}\big(n|\ln(\epsilon)|^2\epsilon^{-2}\big)$ calls to the inexact oracle to find a point whose expected gradient norm is below $\epsilon$, where $n$ is the number of variables of the problem.

As a motivating application, we consider model-free steady-state optimization problems [24, 25, 21, 27]. In this setting, the objective function is defined implicitly through the steady-state behavior of a dynamical system under a constant control input. Since the mapping defining the system dynamics is assumed to be unknown, inexact function evaluations can be obtained by running the system until an approximate steady state is reached. The resulting oracle naturally exhibits controllable inexactness, as the accuracy of the steady-state estimate depends on the simulation or experiment duration. For systems defined by contractions, we show that this oracle satisfies the assumptions of our framework, which illustrate the practical relevance of the proposed method.

The remainder of the paper is organized as follows. Section 2 presents the proposed method along with its convergence and complexity analysis. In Section 3, we describe the application of the proposed method to model-free steady-state optimization problems. Finally, Section 4 presents numerical results illustrating the effectiveness of the proposed method.

# 2 New Method and its Complexity Analysis

We consider the unconstrained optimization problem

$$\min_{u \in \mathbb{R}^n} f(u), \tag{1}$$

where the objective function is specified by the assumptions:

**A1.** $f : \mathbb{R}^n \to \mathbb{R}$ has a global minimizer $u^*$.

**A2.** $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$ is $L$-Lipschitz continuous.

When an exact zeroth-order oracle of $f$ is available, one of the simplest derivative-free methods for problem (1) is the *Stochastic Three Points (STP) method* [2]. The method generates a sequence $\{u_k\}_{k\geq 0}$ according to the rule

$$u_{k+1} = \arg\min \Big\{ f(u_k),\ f(u_k + \alpha_k s_k),\ f(u_k - \alpha_k s_k) \Big\}, \tag{2}$$

where $s_k \sim \mathcal{N}\Big(0, \frac{1}{n} I_n\Big)$ is a random search direction sampled from a standard normal distribution scaled by $1/n$, and $\alpha_k > 0$ is a stepsize, typically chosen by the user according to a predefined sequence. In our setting, however, we assume that only an *inexact* zeroth-order oracle is available, characterized by the following assumption:

**A3.** There exists $C > 0$, such that, for any $(u, \delta) \in \mathbb{R}^n \times \mathbb{R}_{++}$, the oracle returns $f_\delta(u)$ satisfying

$$|f_\delta(u) - f(u)| \leq C\delta.$$

To account for the inexact oracle, we propose the following inexact STP (ISTP) method:

**Algorithm 1.** Stochastic Three Points Method with Inexact Oracle (ISTP)

**Step 0.** Given a starting point $u_0 \in \mathbb{R}^n$, an estimate $\hat{C}$ to constant $C$ in A3, an estimate $\hat{L} > 0$ to the Lipschitz constant $L$ in A2, and a constant $D > 0$, consider the sequence of stepsizes $\{\alpha_k\}_{k\geq 0} \subset \mathbb{R}_{++}$ given by

$$\alpha_k = D/\sqrt{k+1}, \quad k = 0, 1, 2, \ldots, \tag{3}$$

set $k := 0$.

**Step 1.** Define

$$\delta_k = \left(\hat{L}/\hat{C}\right) \frac{\alpha_k^2}{4}, \tag{4}$$

and draw $s_k \sim \mathcal{N}\left(0, \frac{1}{n}I\right)$.

**Step 2.** Compute $f_{\delta_k}(u_k)$, $f_{\delta_k}(u_k + \alpha_k s_k)$ and $f_{\delta_k}(u_k - \alpha_k s_k)$.

**Step 3.** Define

$$u_{k+1} = \arg\min \left\{ f_{\delta_k}(u_k), f_{\delta_k}(u_k + \alpha_k s_k), f_{\delta_k}(u_k - \alpha_k d_k) \right\}, \tag{5}$$

set $k := k + 1$ and go back to Step 1.

In Algorithm 1, the sequence of tolerances $\{\delta_k\}_{k\geq 0}$ is coupled to the stepsizes $\{\alpha_k\}_{k\geq 0}$ through the relation $\delta_k = \mathcal{O}(\alpha_k^2)$ in (4). With the stepsize sequence defined in (3), we get that $\delta_k = \mathcal{O}(1/k)$. This coupling reflects a deliberate balance between exploration and accuracy. In the initial phase, when the stepsizes are relatively large, the method operates with coarser oracle information, which is sufficient to identify promising directions and broadly explore the domain. As the algorithm advances and the stepsizes decrease, higher oracle accuracy is required, ensuring more reliable assessments of candidate points. This gradual refinement enables Algorithm 1 to transition from exploratory behavior to a more cautious search focused on regions where consistent descent of the objective function can be achieved. It is worth noting that other variants of the STP method, based on inexact zeroth-order oracles, have recently been proposed in [33, 4]. However, these works do not consider the oracle defined in our Assumption A3.

The next lemma provides an upper bound on the scaled directional derivative $\alpha_k|\langle \nabla f(u_k), s_k\rangle|$, showing that it is bounded by the decrease in the objective function between successive iterates, up to $\mathcal{O}(\alpha_k^2)$ terms capturing the effect of curvature (Assumption A2) and oracle error (Assumption A3).

**Lemma 2.1.** *Suppose that A2 and A3 hold, and let $\{u_k\}_{k\geq 0}$ be generated by Algorithm 1. Then*

$$\alpha_k|\langle \nabla f(u_k), s_k\rangle| \leq f(u_k) - f(u_{k+1}) + \frac{L}{2}\alpha_k^2\|s_k\|^2 + \left(\frac{C}{\hat{C}}\right)\frac{\hat{L}}{2}\alpha_k^2. \tag{6}$$

*Proof.* By A3 and A2, we have

$$\begin{aligned} f_{\delta_k}(u_k + \alpha_k s_k) &\leq f(u_k + \alpha_k s_k) + C\delta_k \\ &\leq f(u_k) + \alpha_k\langle \nabla f(u_k), s_k\rangle + \frac{L}{2}\alpha_k^2\|s_k\|^2 + C\delta_k. \end{aligned} \tag{7}$$

Now, it follows from (7), (5), A3 and (4) that

$$\begin{aligned}\alpha_k\left[-\langle\nabla f(u_k),s_k\rangle\right] &\leq f(u_k)-f_{\delta_k}(u_k+\alpha_k s_k)+\frac{L}{2}\alpha_k^2\|s_k\|^2+C\delta_k\\ &\leq f(u_k)-f_{\delta_k}(u_{k+1})+\frac{L}{2}\alpha_k^2\|s_k\|^2+C\delta_k\\ &\leq f(u_k)-f(u_{k+1})+\frac{L}{2}\alpha_k^2\|s_k\|^2+2C\delta_k\\ &\leq f(u_k)-f(u_{k+1})+\frac{L}{2}\alpha_k^2\|s_k\|^2+\left(\frac{C}{\hat{C}}\right)\frac{\hat{L}}{2}\alpha_k^2. \end{aligned} \tag{8}$$

On the other hand, we also have

$$\begin{aligned}f_{\delta_k}(u_k-\alpha_k s_k) &\leq f(u_k-\alpha_k s_k)+C\delta_k\\ &\leq f(u_k)-\alpha_k\langle\nabla f(u_k),s_k\rangle+\frac{L}{2}\alpha_k^2\|s_k\|^2+C\delta_k.\end{aligned}$$

Therefore, using the same reasoning as above, we can see that

$$\alpha_k\langle\nabla f(u_k),s_k\rangle\leq f(u_k)-f(u_{k+1})+\frac{L}{2}\alpha_k^2\|s_k\|^2+\left(\frac{C}{\hat{C}}\right)\frac{\hat{L}}{2}\alpha_k^2. \tag{9}$$

Combining (8) and (9), we conclude that (6) is true. □

The next lemma from [2] shows that the random direction $s_k$ is normalized in expectation, and that the expected absolute directional derivative $|\langle\nabla f(u_k),s_k\rangle|$ is proportional to $\|\nabla f(u_k)\|$.

**Lemma 2.2** (Lemma 3.4 in [2])**.** *Let $\{s_k\}_{k\geq 0}\subset\mathbb{R}^n$ be generated by Algorithm 1. Then*

$$E_{s_k}\left(\|s_k\|^2\right)=1 \quad \text{and} \quad E_{s_k}\left(|\langle\nabla f(u_k),s_k\rangle\right)=\frac{\sqrt{2}}{\sqrt{n\pi}}\|\nabla f(u_k)\|.$$

By combining the two previous lemmas, we conclude that Algorithm 1 generates a sequence of iterates such that the minimum of the expected gradient norms over the first $T$ iterations converges to zero.

**Lemma 2.3.** *Suppose that A1-A3 hold and let $\{u_k\}_{k\geq 0}$ be generated by Algorithm 1. Then, given $T\geq 2$, we have*

$$\min_{k=0,\ldots,T-1}E\left(\|\nabla f(u_k)\|\right)\leq\frac{\sqrt{2n\pi}\left(f(u_0)-f(u^*)\right)}{D\sqrt{T}}+\frac{\sqrt{2n\pi}\left[1+\left(\frac{C}{\hat{C}}\right)\left(\frac{\hat{L}}{L}\right)\right]LD\ln(T)}{\sqrt{T}}. \tag{10}$$

*Consequently,*

$$\lim_{T\rightarrow+\infty}\min_{k=0,\ldots,T-1}E\left(\|\nabla f(u_k)\|\right)=0. \tag{11}$$

*Proof.* From Lemma 2.1 we have

$$\alpha_k|\langle\nabla f(u_k),s_k\rangle|\leq f(u_k)-f(u_{k+1})+\frac{L}{2}\alpha_k^2\|s_k\|^2+\left(\frac{C}{\hat{C}}\right)\left(\frac{\hat{L}}{L}\right)\frac{L}{2}\alpha_k^2$$

Then, taking the expectation with respect to $s_k$ on both sides, and using Lemma 2.2, we obtain

$$\alpha_k \frac{\sqrt{2}}{\sqrt{n\pi}} \|\nabla f(u_k)\| \leq f(u_k) - E_{s_k}(f(u_{k+1})) + \frac{1}{2}\left[1 + \left(\frac{C}{\hat{C}}\right)\left(\frac{\hat{L}}{L}\right)\right] L\alpha_k^2.$$

Now, taking the expectation with respect to $s_0, \ldots, s_k$, it follows that

$$\alpha_k \frac{\sqrt{2}}{\sqrt{n\pi}} E\left(\|\nabla f(u_k)\|\right) \leq E(f(u_k)) - E(f(u_{k+1})) + \frac{1}{2}\left[1 + \left(\frac{C}{\hat{C}}\right)\left(\frac{\hat{L}}{L}\right)\right] L\alpha_k^2.$$

Summing up these inequalities for $k = 0, \ldots, T-1$, and using A1, we have

$$\begin{aligned}
\left(\sum_{k=0}^{T-1} \alpha_k\right) \frac{\sqrt{2}}{\sqrt{n\pi}} \min_{k=0,\ldots,T-1} E\left(\|\nabla f(u_k)\|\right) &\leq \sum_{k=0}^{T-1} \alpha_k \frac{\sqrt{2}}{\sqrt{n\pi}} E\left(\|\nabla f(u_k)\|\right) \\
&\leq E(f(u_0)) - E(f(u_T)) + \frac{1}{2}\left[1 + \left(\frac{C}{\hat{C}}\right)\left(\frac{\hat{L}}{L}\right)\right] L\left(\sum_{k=0}^{T-1} \alpha_k^2\right) \\
&\leq f(u_0) - f(u^*) + \frac{1}{2}\left[1 + \left(\frac{C}{\hat{C}}\right)\left(\frac{\hat{L}}{L}\right)\right] L\left(\sum_{k=0}^{T-1} \alpha_k^2\right). \qquad (12)
\end{aligned}$$

Note that, since $T \geq 2$, we have

$$\sum_{k=0}^{T-1} \alpha_k = \sum_{k=0}^{T-1} \frac{D}{\sqrt{k+1}} \geq \frac{D}{2}\sqrt{T}. \qquad (13)$$

Moreover,

$$\sum_{k=0}^{T-1} \alpha_k^2 = \sum_{k=0}^{T-1} \frac{D^2}{\sqrt{k+1}} \leq 2D^2 \ln(T). \qquad (14)$$

Combining (12)-(14), it follows that

$$\frac{D}{2}\sqrt{T} \frac{\sqrt{2}}{\sqrt{n\pi}} \min_{k=0,\ldots,T-1} E\left(\|\nabla f(u_k)\|\right) \leq f(u_0) - f(u^*) + \left[1 + \left(\frac{C}{\hat{C}}\right)\left(\frac{\hat{L}}{L}\right)\right] LD^2 \ln(T).$$

which implies that (10) is true. Finally, using the fact that $\lim_{T\to+\infty} \frac{\ln(T)}{\sqrt{T}} = \lim_{T\to+\infty} \frac{1}{\sqrt{T}} = 0$ in (10), we conclude that (11) is true. $\square$

Given $\epsilon > 0$, define

$$T(\epsilon) = \inf\left\{k \in \mathbb{N} \,:\, E\big(\|\nabla f(u_k)\|\big) \leq \epsilon\right\}, \qquad (15)$$

where $E(\cdot)$ denotes the (unconditional) expectation with respect to all the randomness generated by the algorithm up to iteration $k$, i.e., with respect to $s_0, \ldots, s_{k-1}$. From Lemma 2.3, we will show that Algorithm 1 requires at most $\mathcal{O}\big(n|\ln(\epsilon)|^2\epsilon^{-2}\big)$ iterations to find a point whose expected gradient norm is less than or equal to $\epsilon$. For that, let us consider the following auxiliary result.

**Lemma 2.4.** *Given $b \in \big(0, \frac{2}{e^2}\big]$, let*

$$a = \frac{16\left[\ln\left(\frac{2}{b}\right)\right]^2}{b^2}. \qquad (16)$$

*Then*

$$\frac{\ln(T)}{\sqrt{T}} \leq b, \quad \text{for all} \quad T \geq a. \qquad (17)$$

*Proof.* Consider the univariate function $h(x) = \frac{\ln x}{\sqrt{x}}$. A direct computation shows that

$$h'(x) = \frac{2 - \ln x}{2x^{3/2}},$$

so $h$ is decreasing for all $x > e^2$. Since $b \in (0, 2/e^2]$, we have $\ln(2/b) \geq 2$, and therefore

$$a = \frac{16\left[\ln(\frac{2}{b})\right]^2}{b^2} > 16 > e^2.$$

Thus, $h$ is decreasing on $[a, \infty)$, and to prove (17) it suffices to prove that $h(a) \leq b$, i.e.,

$$\ln(a) \leq b\sqrt{a}. \tag{18}$$

From the definition of $a$, we compute

$$\ln(a) = \ln\left(\frac{16\,[\ln(2/b)]^2}{b^2}\right) = \ln 16 + 2\ln\big(\ln(2/b)\big) - 2\ln b,$$

and

$$\sqrt{a} = \frac{4\ln(2/b)}{b}.$$

Hence, the inequality $\ln(a) \leq b\sqrt{a}$ is equivalent to

$$\ln 16 + 2\ln\big(\ln(2/b)\big) - 2\ln b \leq 4\ln(2/b).$$

Using $\ln 16 = 4\ln 2$ and $\ln(2/b) = \ln 2 - \ln b$, this simplifies to

$$2\ln\big(\ln(2/b)\big) \leq -2\ln b,$$

or equivalently,

$$\ln\big(\ln(2/b)\big) \leq -\ln b = \ln b^{-1}.$$

Since the logarithm is increasing, we conclude that (18) is equivalent to

$$\ln(2/b) \leq \frac{1}{b}.$$

To prove this, define $p(z) = \frac{1}{z} - \ln(2/z)$. Then

$$p'(z) = \frac{z-1}{z^2} < 0 \quad \text{for } z \in (0,1),$$

so $p$ is decreasing on $(0, 1]$, and in particular on $(0, 2/e^2]$. Thus,

$$p(b) \geq p(2/e^2) = \frac{e^2}{2} - \ln(e^2) = \frac{e^2}{2} - 2 > 0,$$

which shows that $\ln(2/b) < 1/b$. Thus, we have

$$\ln(2/b) < 1/b \Longrightarrow \ln(a) \leq b\sqrt{a} \Longrightarrow h(a) \leq b,$$

and since $h$ is decreasing on $[a, \infty)$, it follows that (17) is true for every $T \geq a$. $\square$

Now we are ready to establish a worst-case complexity bound for Algorithm 1.

**Theorem 2.5.** *Suppose that A1-A3 hold and let $\{u_k\}_{k\geq 0}$ be generated by Algorithm 1. Given*

$$0 < \epsilon \leq \left(\frac{4}{e^2}\right)\sqrt{2\pi n}\left[1+\left(\frac{C}{\hat{C}}\right)\left(\frac{\hat{L}}{L}\right)\right]LD \tag{19}$$

*let $T(\epsilon)$ be defined in (15). Then*

$$T(\epsilon) \leq \max\left\{1, 8n\pi\left(\frac{f(u_0)-f(u^*)}{D}\right)^2\epsilon^{-2}, (128n\pi)G^2L^2D^2\left[\ln\left(\frac{4\sqrt{2n\pi}GLD}{\epsilon}\right)\right]^2\epsilon^{-2}\right\}, \tag{20}$$

*where*

$$G := \left[1+\left(\frac{C}{\hat{C}}\right)\left(\frac{\hat{L}}{L}\right)\right]. \tag{21}$$

*Proof.* Assume by contradiction that

$$T(\epsilon) > \max\left\{1, 8n\pi\left(\frac{f(u_0)-f(u^*)}{D}\right)^2\epsilon^{-2}, (128n\pi)G^2L^2D^2\left[\ln\left(\frac{4\sqrt{2n\pi}GLD}{\epsilon}\right)\right]^2\epsilon^{-2}\right\}. \tag{22}$$

In view of (19), we have

$$b = \frac{\epsilon}{2\sqrt{2n\pi}GLD} \in (0, 2/e^2]$$

Thus, we can apply Lemma 2.4 with

$$a = \frac{16[\ln(2/b)]^2}{b^2} = (128n\pi)G^2L^2D^2\left[\ln\left(\frac{4\sqrt{2n\pi}GLD}{\epsilon}\right)\right]^2\epsilon^{-2}.$$

Specifically, due to (22), it follows that

$$\frac{\ln(T(\epsilon))}{\sqrt{T(\epsilon)}} \leq b = \frac{\epsilon}{2\sqrt{2n\pi}GLD}$$

which, by (21), means that

$$\frac{\sqrt{2n\pi}\left[1+\left(\frac{C}{\hat{C}}\right)\left(\frac{\hat{L}}{L}\right)\right]LD\ln(T(\epsilon))}{\sqrt{T(\epsilon)}} \leq \frac{\epsilon}{2}. \tag{23}$$

On the other hand, by (22) we also have

$$\frac{\sqrt{2n\pi}\left(f(u_0)-f(u^*)\right)}{D\sqrt{T(\epsilon)}} < \frac{\epsilon}{2} \tag{24}$$

However, in this case, applying inequality (10) of Lemma 2.3 with $T = T(\epsilon)$, we would get the contradiction

$$\epsilon < \min_{k=0,\ldots,T(\epsilon)-1} E\left(\|\nabla f(u_k)\|\right) \leq \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon.$$

Therefore, (22) must be false. □

**Remark 2.6.** *Since every iteration of Algorithm 1 requires three calls to the oracle, Theorem 2.5 establishes that Algorithm 1 takes no more than $\tilde{\mathcal{O}}(n\epsilon^{-2})$ calls to the inexact oracle to find $u_k$ such that $E(\|\nabla f(u_k)\|) \leq \epsilon$, where $\tilde{\mathcal{O}}(\cdot)$ hides logarithmic factors in $\epsilon$.*

## 3 Application to Model-Free Steady-State Optimization

Consider the discrete-time dynamical system

$$\begin{cases} x_{t+1}(u) &= \varphi(x_t(u), u), \;\; t = 0, 1, 2, \ldots, \\ x_0(u) &= x_0, \end{cases} \tag{25}$$

where $u \in \mathbb{R}^n$ is a constant input control, $x_t(u) \in \mathbb{R}^p$ is the state at time $t$, and the dynamics is defined by the function $\varphi : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}^p$. In what follows, we will assume that:

**A4.** For every $u \in \mathbb{R}^n$, function $\varphi(\,\cdot\,, u)$ is $L_\varphi$-Lipschitz continuous with $L_\varphi < 1$.

**Example 3.1.** *Given a control $u \in \mathbb{R}^n$, consider the linear dynamical system*

$$x_{t+1}(u) = Ax_t(u) + Bu + d,$$

*with $A \in \mathbb{R}^{p\times p}$, $B \in \mathbb{R}^{p\times n}$, $d \in \mathbb{R}^p$, and $\|A\| < 1$. This system can be written in the form (25) with functions $\varphi(x, u) = Ax + Bu + d$. In addition, function $\varphi(\,\cdot\,, u)$ satisfies assumption A4 with Lipschitz constant $L_\varphi = \|A\|$.*

In view of Assumption A4, it follows from the Banach Fixed Point Theorem that, for every $u \in \mathbb{R}^n$, there exists a unique point $x^*(u) \in \mathbb{R}^p$ such that

$$x^*(u) = \varphi(x^*(u), u), \tag{26}$$

and $\lim_{t\to+\infty} x_t(u) = x^*(u)$. The point $x^*(u)$ is referred to as the steady state of system (25). In this context, we consider the following steady-state optimization problem:

$$\min_{u\in\mathbb{R}^n} \; f(u) \equiv F(x^*(u), u), \tag{27}$$

where $F : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}$ is a differentiable, nonnegative cost function, and $x^*(u)$ denotes the steady state of the dynamical system (25) corresponding to the constant input $u$. Specifically, we will assume that:

**A5.** For every $u \in \mathbb{R}^n$, the function $F(\,\cdot\,, u) : \mathbb{R}^p \to \mathbb{R}$ is $L_F$-Lipschitz continuous.

**Example 3.2.** *Given $\bar{x} \in \mathbb{R}^p$, $\lambda > 0$ and $\mu > 0$, consider the function $F : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}$ given by*

$$F(x, u) = H_\mu(x - \bar{x}) + \lambda\Phi(u),$$

*where*

$$\Phi(u) = \sum_{i=1}^{n} \frac{u_i^2}{1 + u_i^2}, \quad \forall u \in \mathbb{R}^n,$$

*and*

$$H_\mu(z) = \sum_{j=1}^{p} h_\mu(z_i), \quad \forall z \in \mathbb{R}^p,$$

*with $h_\mu : \mathbb{R} \to \mathbb{R}$ being the Huber function with parameter $\mu$ defined by*

$$h_\mu(z_j) = \begin{cases} \frac{1}{2\mu}|z_j|^2, & \textit{if } |z_j| \leq \mu, \\ \\ |z_j| - \frac{\mu}{2}, & \textit{otherwise.} \end{cases}$$

*Since*

$$\|\nabla_x F(x,u)\| = \|\nabla H_\mu(x-\bar{x})\| \leq \sqrt{p}, \quad \forall x \in \mathbb{R}^p,$$

*it follows that, for every $u \in \mathbb{R}^n$, the function $F(\,\cdot\,,u)$ above satisfies assumption A5 with Lipschitz constant $L_F = \sqrt{p}$.*

Steady-state optimization problems of the form (27) arise in several control applications, including the operation of continuous stirred-tank reactors [17], building heating and thermal regulation systems [35], and control of power systems [6]. When the function $\varphi$ in (25) is unknown, the mapping $u \mapsto x^*(u)$ cannot be computed explicitly. Consequently, neither the exact function values nor the gradients of $f(\,\cdot\,)$ in (27) are directly accessible (see, e.g., [24, 25, 21, 27]). In this *model-free* setting, however, it is possible to approximate $f(u)$ by running the underlying dynamical system for a finite number of time steps, thereby obtaining an approximation of the steady state. More precisely, one can iterate the system until a prescribed tolerance is met, i.e.,

$$\|x_{t+1}(u) - x_t(u)\| \leq \delta,$$

for some $\delta > 0$. The resulting iterate $x_{t+1}(u)$ is then an approximation of the true steady state $x^*(u)$, and $F(x_{t+1}(u),u)$ provides a corresponding approximation of $f(u)$.

The next two lemmas show that this procedure yields an inexact zeroth-order oracle for $f(\,\cdot\,)$, which satisfies Assumption A3.

**Lemma 3.3.** *Suppose that A4 holds and let $\{x_t(u)\}_{t\geq 0}$ be generated by (25). Given $\delta > 0$, if*

$$\|x_{t+1}(u) - x_t(u)\| \leq \delta \tag{28}$$

*for some $t \geq 0$, then*

$$\|x_{t+1}(u) - x^*(u)\| \leq \left(\frac{L_\varphi}{1-L_\varphi}\right)\delta.$$

*Proof.* By (28), (25), (26), and Assumption A4, we have

$$\begin{aligned}\|x_t(u) - x^*(u)\| &\leq \|x_t(u) - x_{t+1}(u)\| + \|x_{t+1}(u) - x^*(u)\| \leq \delta + \|\varphi(x_t(u);u) - \varphi(x^*(u);u)\| \\ &\leq \delta + L_\varphi \|x_t(u) - x^*(u)\|.\end{aligned}$$

Thus,

$$\|x_t(u) - x^*(u)\| \leq \frac{\delta}{1-L_\varphi},$$

and so

$$\|x_{t+1}(u) - x^*(u)\| = \|\varphi(x_t(u);u) - \varphi(x^*(u);u)\| \leq L_\varphi \|x_t(u) - x^*(u)\| \leq \left(\frac{L_\varphi}{1-L_\varphi}\right)\delta.$$

□

**Lemma 3.4.** *Suppose that A4-A5 hold and let $\{x_t(u)\}_{t\geq 0}$ be generated by (25). Given $\delta > 0$, if (28) holds for some $t \geq 0$, then*

$$|F(x_{t+1}(u),u) - f(u)| \leq L_F \left(\frac{L_\varphi}{1-L_\varphi}\right)\delta,$$

*where $f(\,\cdot\,)$ is defined in (27).*

*Proof.* Indeed, it follows directly from (27), A5 and Lemma 3.3:

$$\begin{aligned}|F(x_{t+1}(u),u)-f(u)| &= |F(x_{t+1}(u),u)-F(x^*(u),u)| \leq L_F\|x_{t+1}(u)-x^*(u)\| \\ &\leq L_F\left(\frac{L_\varphi}{1-L_\varphi}\right)\delta.\end{aligned}$$

□

Lemmas 3.3 and 3.4 motivate the following *Run-to-Quasi-Steady-State* procedure to compute an approximation $f_\delta(u)$ of $f(u)$.

**Algorithm 2.** $(f_\delta(u), x^+(u)) = \texttt{InexactOracle}\,(u, x_0, \delta)$

**Step 0.** Set $x_0(u) = x_0$ and $t := 0$.

**Step 1.** Evaluate the state

$$x_{t+1}(u) = \varphi(x_t(u); u).$$

**Step 2.** If $\|x_{t+1}(u) - x_t(u)\| \leq \delta$, stop and return $x^+(u) = x_{t+1}(u)$ and $f_\delta(u) = F(x^+(u), u)$. Otherwise, set $t := t+1$ and go back to Step 1.

**Remark 3.5.** *By Lemma 3.4, the output $f_\delta(u)$ of Algorithm 2 satisfies the error bound*

$$|f_\delta(u) - f(u)| \leq C\delta,$$

*where*

$$C = L_F\left(\frac{L_\varphi}{1-L_\varphi}\right). \tag{29}$$

*Therefore, Algorithm 2 provides an inexact oracle for $f(\,\cdot\,)$ in (27) that satisfies Assumption A3.*

The iterations of Algorithm 2 will be referred to as *time steps*, in order to distinguish them from the iterations of the optimization algorithm. The next lemma establishes an upper bound of $\mathcal{O}\left(\log(\delta^{-1})\right)$ on the number of time steps required by Algorithm 2 to terminate and return $f_\delta(u)$.

**Lemma 3.6.** *Suppose that A4 hold and let $\{x_t(u)\}_{t\geq 0}$ be generated by (25) with $x_0 \neq x^*(u)$. Given $\delta > 0$, define*

$$t(\delta) = \inf\left\{t \in \mathbb{N}\,:\,\|x_{t+1}(u) - x_t(u)\| \leq \delta\right\}. \tag{30}$$

*Then*

$$t(\delta) \leq 2 + \frac{\log\left(2\|x_0 - x^*(u)\|\delta^{-1}\right)}{|\log(L_\varphi)|}. \tag{31}$$

*Proof.* If $t(\delta) \leq 1$, then (31) is clearly true. Thus, suppose that $t(\delta) \geq 2$. Then, in view of (30), (25) and A4, we have

$$\begin{aligned}\delta &< \|x_{t(\delta)}(u) - x_{t(\delta)-1}(u)\| \leq \|x_{t(\delta)}(u) - x^*(u)\| + \|x_{t(\delta)-1}(u) - x^*(u)\| \\ &\leq L_\varphi^{t(\delta)}\|x_0 - x^*(u)\| + L_\varphi^{t(\delta)-1}\|x_0 - x^*(u)\| \leq 2L_\varphi^{t(\delta)-1}\|x_0 - x^*(u)\|.\end{aligned}$$

That is,

$$\frac{\delta}{2\|x_0 - x^*(u)\|} \leq L_\varphi^{t(\delta)-1}.$$

Taking the logarithm in both sides of this inequality, we obtain

$$\log\left(\frac{\delta}{2\|x_0 - x^*(u)\|}\right) \leq (t(\delta) - 1)\log(L_\varphi).$$

Since $L_\varphi \in (0, 1)$, dividing both sides by $\log(L_\varphi)$ with see that $t(\delta)$ satisfies (31). □

Now, assuming that $f(\cdot)$ in (27) satisfies A1 and A2, we can specialize Algorithm 1 to problem (27):

**Algorithm 3.** ISTP for Model-Free Steady-State Optimization

**Step 0.** Given a starting point $u_0 \in \mathbb{R}^n$, an estimate $\hat{C}$ to constant $C$ in (29), an estimate $\hat{L} > 0$ to the Lipschitz constant $L$ of $\nabla f(\cdot)$, and a constant $D > 0$, consider the sequence of stepsizes $\{\alpha_k\}_{k\geq 0} \subset \mathbb{R}_{++}$ given by

$$\alpha_k = D/\sqrt{k+1}, \quad k = 0, 1, 2, \ldots$$

Denoting by $x_0$ the initial state of the system, set $\hat{x}_0 = x_0$ and $k := 0$.

**Step 1.** Define

$$\delta_k = \left(\hat{L}/\hat{C}\right)\frac{\alpha_k^2}{4},$$

and draw $s_k \sim \mathcal{N}\left(0, \frac{1}{n}I\right)$.

**Step 2.** Call Algorithm 2, and evaluate in this order:

$$\begin{aligned}
\left(f_{\delta_k}(u_k), x^+(u_k)\right) &= \texttt{InexactOracle}\left(u_k, \hat{x}_k, \delta_k\right), \\
\left(f_{\delta_k}(u_k + \alpha_k s_k), x^+(u_k + \alpha_k s_k)\right) &= \texttt{InexactOracle}\left(u_k + \alpha_k s_k, x^+(u_k), \delta_k\right), \\
\left(f_{\delta_k}(u_k - \alpha_k s_k), x^+(u_k - \alpha_k s_k)\right) &= \texttt{InexactOracle}\left(u_k - \alpha_k s_k, x^+(u_k + \alpha_k s_k), \delta_k\right).
\end{aligned}$$

**Step 3.** Define $\hat{x}_{k+1} = x^+(u_k - \alpha_k s_k)$ and

$$u_{k+1} = \arg\min\left\{f_{\delta_k}(u_k), f_{\delta_k}(u_k + \alpha_k s_k), f_{\delta_k}(u_k - \alpha_k d_k)\right\}.$$

Set $k := k + 1$ and go back to Step 1.

**Remark 3.7.** *Step 2 of Algorithm 3 requires three calls to Algorithm 2 with*

$$\delta_k = \left(\frac{\hat{L}}{\hat{C}}\right)\frac{\alpha_k^2}{4} = \left(\frac{\hat{L}}{\hat{C}}\right)\frac{D^2}{4(k+1)}.$$

*Thus, due to Lemma 3.6, the number $t_k$ of time steps executed at the kth iteration of Algorithm 3 is bounded by*

$$t_k \leq 3\left[2 + \frac{1}{|\log(L_\varphi)|}\log\left(\frac{8\left(\frac{\hat{C}}{\hat{L}}\right)M_k}{D^2}(k+1)\right)\right], \tag{32}$$

*where*

$$M_k = \max\Big\{\|\hat{x}_k - x^*(u_k)\|,\ \|x^+(u_k) - x^*(u_k + \alpha_k s_k)\|,\ \|x^+(u_k + \alpha_k s_k) - x^*(u_k - \alpha_k s_k)\|\Big\}. \tag{33}$$

The next example presents a steady-state optimization problem that satisfies all the assumptions used in our analysis.

**Example 3.8.** *Consider problem (27) with $F(\,\cdot\,,\,\cdot\,)$ given in Example 3.2 and $x^*(u)$ being the steady state of the dynamical system described in Example 3.1, with matrix $B$ having full column rank. In this case, we have*

$$x^*(u) = (I-A)^{-1}(Bu+d),$$

*and thus the objective function in (27) can be written as*

$$f(u) = F(x^*(u),u) = H_\mu\big((I-A)^{-1}(Bu+d)-\bar{x}\big) + \lambda\Phi(u). \tag{34}$$

*Since $H_\mu(\,\cdot\,)$ is coercive, $\lambda\Phi(\,\cdot\,)$ is nonnegative, and $B$ has full column rank, it follows that*

$$\lim_{\|u\|\to+\infty} f(u) = +\infty,$$

*i.e., $f(\,\cdot\,)$ is coercive. Therefore, $f(\,\cdot\,)$ admits a global minimizer $u^*$, and Assumption A1 is satisfied. From (34), we obtain*

$$\nabla f(u) = \big[(I-A)^{-1}B\big]^T \nabla H_\mu\big((I-A)^{-1}(Bu+d)-\bar{x}\big) + \lambda\nabla\Phi(u).$$

*For the Huber function $H_\mu(\,\cdot\,)$, it follows that $\nabla H_\mu(\,\cdot\,)$ is Lipschitz continuous with constant $L_H = \mu^{-1}$. Moreover, since*

$$\big\|\nabla^2\Phi(u)\big\| = \max_{i=1,\ldots,n}\left|\frac{d^2}{dz_i^2}\left(\frac{z_i^2}{1+z_i^2}\right)\right| < 6, \quad \forall u \in \mathbb{R}^n,$$

*it follows that $\nabla\Phi(\,\cdot\,)$ is Lipschitz continuous with constant $L_\Phi = 6$. Hence, for any $u, w \in \mathbb{R}^n$,*

$$\begin{aligned}
\|\nabla f(u) - \nabla f(w)\| &\leq \big\|(I-A)^{-1}B\big\|\,\big\|\nabla H_\mu\big((I-A)^{-1}(Bu+d)-\bar{x}\big) - \nabla H_\mu\big((I-A)^{-1}(Bw+d)-\bar{x}\big)\big\| \\
&\quad +\lambda\|\nabla\Phi(u)-\nabla\Phi(w)\| \\
&\leq \mu^{-1}\big\|(I-A)^{-1}B\big\|\,\big\|(I-A)^{-1}B(u-w)\big\| + 6\lambda\|u-w\| \\
&\leq \Big[\mu^{-1}\big\|(I-A)^{-1}B\big\|^2 + 6\lambda\Big]\,\|u-w\|.
\end{aligned}$$

*Thus, $f(\,\cdot\,)$ satisfies Assumption A2 with Lipschitz constant*

$$L = \mu^{-1}\big\|(I-A)^{-1}B\big\|^2 + 6\lambda.$$

*Therefore, this example satisfies Assumptions A1, A2, A4, and A5.*

In view of Remark 3.5, Algorithm 2 provides an inexact zeroth-order oracle for $f(\,\cdot\,)$ that satisfies Assumption A3, with the constant $C$ given by (29). Consequently, Theorem 2.5 immediately implies that Algorithm 3 achieves a worst-case iteration complexity of $\tilde{\mathcal{O}}\big(n\epsilon^{-2}\big)$ to find, in expectation, an $\epsilon$-approximate stationary point of $f(\,\cdot\,)$ in (27).

**Theorem 3.9.** *Consider the optimization problem defined by (27) and (26), and suppose that assumptions A1, A2, A4 and A5 hold. Let $\{u_k\}_{k\geq 0}$ be a sequence generated by Algorithm 3 applied to (27). Given*

$$0 < \epsilon \leq \left(\frac{4}{e^2}\right)\sqrt{2n\pi}\left[1 + \frac{L_F\left(\frac{L_\varphi}{1-L_\varphi}\right)}{\hat{C}}\left(\frac{\hat{L}}{L}\right)\right]LD \tag{35}$$

*let $T(\epsilon)$ be defined in (15). Then, we have*

$$T(\epsilon) \leq \max\left\{1, 8n\pi\left(\frac{f(u_0)-f(u^*)}{D}\right)^2\epsilon^{-2}, (128n\pi)G^2L^2D^2\left[\ln\left(\frac{4\sqrt{2n\pi}GLD}{\epsilon}\right)\right]^2\epsilon^{-2}\right\}, \tag{36}$$

*where*

$$G := \left[1+\left(\frac{L_F\left(\frac{L_\varphi}{1-L_\varphi}\right)}{\hat{C}}\right)\left(\frac{\hat{L}}{L}\right)\right]. \tag{37}$$

Finally, by combining Theorem 3.9 and Remark 3.7, we can establish an upper bound on the total number of time steps of the dynamical system (25) required by Algorithm 3 to find, in expectation, an $\epsilon$-approximate stationary point of $f(\,\cdot\,)$.

**Theorem 3.10.** *Under the same assumptions as in Theorem 3.9, the total number $\tau(\epsilon)$ of time steps required by Algorithm 3 to find $u_k$ such that $E\big[\|\nabla f(u_k)\|\big] \leq \epsilon$ is bounded from above by*

$$\tau(\epsilon) \leq 3\left[2+\frac{1}{|\log(L_\varphi)|}\log\left(\frac{8\left(\frac{\hat{C}}{\hat{L}}\right)M(\epsilon)}{D^2}T(\epsilon)\right)\right]T(\epsilon),$$

*where*

$$M(\epsilon) = \max_{k=0,\ldots,T(\epsilon)-1} M_k, \tag{38}$$

*and $M_k$ is defined in (33).*

*Proof.* By Remark 3.7, the number of time steps $t_k$ executed at the $k$th iteration of Algorithm 3 to compute $f_{\delta_k}(u_k)$, $f_{\delta_k}(u_k+\alpha_k s_k)$, and $f_{\delta_k}(u_k-\alpha_k s_k)$ satisfies

$$t_k \leq 3\left[2+\frac{1}{|\log(L_\varphi)|}\log\left(\frac{8\left(\frac{\hat{C}}{\hat{L}}\right)M_k}{D^2}(k+1)\right)\right].$$

Hence, the total number of time steps up to iteration $T(\epsilon)-1$, denoted $t(\epsilon)$, is bounded by

$$\begin{aligned}\tau(\epsilon) = \sum_{k=0}^{T(\epsilon)-1} t_k &\leq \sum_{k=0}^{T(\epsilon)-1} 3\left[2+\frac{1}{|\log(L_\varphi)|}\log\left(\frac{8\left(\frac{\hat{C}}{\hat{L}}\right)M_k}{D^2}(k+1)\right)\right] \\ &\leq \sum_{k=0}^{T(\epsilon)-1} 3\left[2+\frac{1}{|\log(L_\varphi)|}\log\left(\frac{8\left(\frac{\hat{C}}{\hat{L}}\right)M(\epsilon)}{D^2}T(\epsilon)\right)\right] \\ &= 3\left[2+\frac{1}{|\log(L_\varphi)|}\log\left(\frac{8\left(\frac{\hat{C}}{\hat{L}}\right)M(\epsilon)}{D^2}T(\epsilon)\right)\right]T(\epsilon).\end{aligned}$$

This establishes the stated upper bound. □

**Remark 3.11.** *In view of Theorems 3.9 and 3.10, Algorithm 3 requires at most $\tilde{O}(n\epsilon^{-2})$ time steps of the dynamical system (25) to find $u_k$ such that $E\big[\|\nabla f(u_k)\|\big] \leq \epsilon$.*

# 4 Illustrative Numerical Results

In this section, we present preliminary numerical results illustrating the performance of ISTP for model-free steady-state optimization (Algorithm 3). Given $\gamma \in (0,1)$, we consider the linear system (consistent with Example 3.1)

$$x_{t+1}(u) = A(\gamma)\, x_t(u) + Bu + d, \tag{39}$$

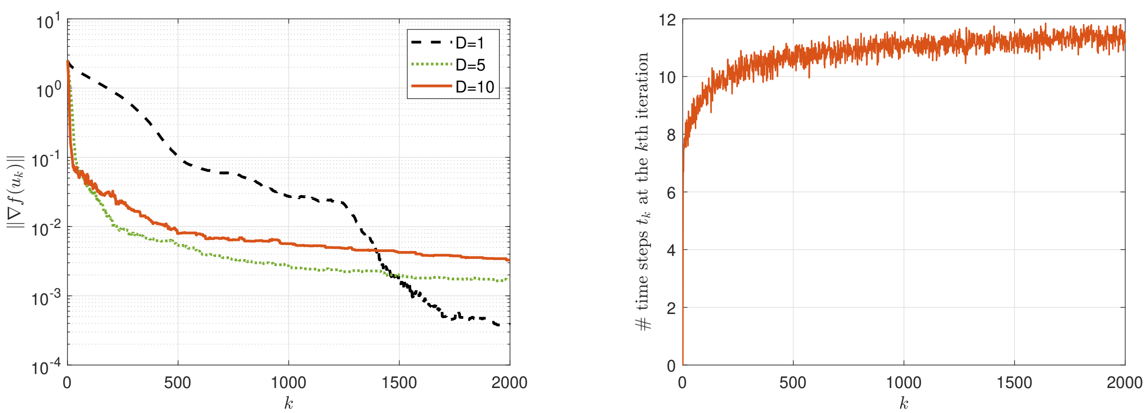
where $A(\gamma) = \frac{\gamma}{\|G\|} G \in \mathbb{R}^{10\times 10}$, $B \in \mathbb{R}^{10\times 5}$, and $d \in \mathbb{R}^{10}$. The entries of $G$, $B$, and $d$ are independently drawn from the standard uniform distribution $\mathcal{U}(0,1)$. By construction, $\|A(\gamma)\| = \gamma$. We consider the following family of optimization problems:

$$\min_{u\in\mathbb{R}^5} \; f_\gamma(u) \equiv H_\mu\big(x^*(u) - \bar{x}\big) + \lambda\Phi(u), \tag{40}$$

where the functions $H_\mu(\cdot)$ and $\Phi(\cdot)$ are defined in Example 3.2, and $x^*(u) = (I - A(\gamma))^{-1}(Bu + d)$ denotes the steady state of (39) associated with the control input $u \in \mathbb{R}^5$. The target steady state $\bar{x} \in \mathbb{R}^{10}$ is defined as

$$\bar{x} = (I - A(\gamma))^{-1}(B\bar{u} + d), \tag{41}$$

where $\bar{u} = \begin{bmatrix}10 & 0 & 10 & 0 & 10\end{bmatrix}^T$. The version of Algorithm 3 (ISTP) tested was the one with parameters $\hat{C} = \sqrt{10}\left(\frac{0.9}{0.1}\right)$ and $\hat{L} = \frac{\hat{C}}{D}$. In all tests we used $u_0 = \begin{bmatrix}0 & 0 & 0 & 0 & 0\end{bmatrix}^T$ as starting point and $x_0 = (I - A(\gamma))^{-1}d$ as initial state. Experiments were conducted in MATLAB R2021a on a 64-bit Windows machine with an Intel Core i7-1165G7 processor and 32 GB RAM.




(a) Average $\|\nabla f(u_k)\|$ vs iteration $k$.

(b) Number of time steps $t_k$ at the $k$th iteration.

Figure 1: Illustrative behavior of ISTP, averaged over 20 runs. Left: convergence of the stationarity measure. Right: evolution of the number of time steps per iteration for $D = 10$.

Figure 1 reports the behavior of the proposed method, applied to (40) with $\gamma = 0.1$, $\mu = 10^2$ and $\lambda = 0.1$. Each curve is obtained by averaging over 20 independent runs. The left plot (Figure 1(a)) shows the evolution of the average stationarity measure $\|\nabla f(u_k)\|$ as a function of the iteration index $k$ for $D \in \{1, 5, 10\}$. For the three choices of $D$, we observe a clear decay of $\|\nabla f(u_k)\|$, illustrating the convergence behavior predicted by Lemma 2.3. The right plot (Figure 1(b)) reports the evolution of the number of time steps $t_k$ as a function of $k$, in the case $D = 10$. The result indicate that $t_k$ grows logarithmically with $k$, in agreement with inequality (32) derived from Lemma 3.6.

Table 1 reports the results of ISTP applied to problem (40) with $\mu = 10^2$, $\lambda = 0$, $\gamma \in \{0.1, 0.3, 0.6, 0.9\}$, and $D \in \{1, 5, 10\}$. The results are averaged over 20 independent runs with a budget of TS = 50,000 time steps. IT denotes the average number of iterations, and TS/IT the average number of time steps per iteration. The last two columns report the average norm of the gradient and the average function value at the final iterate. As $\gamma$ increases, the cost per iteration rises significantly, which leads to higher TS/IT values and fewer iterations within the fixed time budget. This reduction in iterations generally results in larger gradient norms and higher final function values. For $\gamma = 0.1$, all values of $D$ give very similar results, with low gradient norms and function values. When $\gamma = 0.3$, the effect of $D$ becomes clearer, and smaller values of $D$ lead to better convergence. For larger values of $\gamma$ (0.6 and 0.9), the behavior is less uniform: while $D = 1$ allows more iterations, it does not always produce the best results, and intermediate values such as $D = 5$ often give lower gradient norms and function values. Overall, the method performs well for small $\gamma$, but for larger $\gamma$, the choice of $D$ has a stronger impact on the results.

Table 1: Performance of ISTP for different values of $\gamma$ and $D$.

| $\|A(\gamma)\| = \gamma = 0.1$ | | | | |
|---|---|---|---|---|
| $D$ | IT | TS/IT | Average $\|\nabla f(u_{\text{final}})\|$ | Average $f(u_{\text{final}})$ |
| 1 | 4555 | 10.97 | 3.852277E-04 | 5.522718E-07 |
| 5 | 4437 | 11.27 | 6.091647E-04 | 2.919808E-06 |
| 10 | 4432 | 11.28 | 9.163947E-04 | 1.098715E-05 |
| $\|A(\gamma)\| = \gamma = 0.3$ | | | | |
| $D$ | IT | TS/IT | Average $\|\nabla f(u_{\text{final}})\|$ | Average $f(u_{\text{final}})$ |
| 1 | 3099 | 16.13 | 4.236447E-04 | 5.916292E-07 |
| 5 | 2977 | 16.79 | 8.455342E-04 | 5.386328E-06 |
| 10 | 2983 | 16.76 | 1.606957E-03 | 1.771162E-05 |
| $\|A(\gamma)\| = \gamma = 0.6$ | | | | |
| $D$ | IT | TS/IT | Average $\|\nabla f(u_{\text{final}})\|$ | Average $f(u_{\text{final}})$ |
| 1 | 1745 | 28.65 | 1.869532E-03 | 2.526982E-04 |
| 5 | 1606 | 31.13 | 6.203074E-03 | 1.501066E-02 |
| 10 | 1607 | 31.11 | 5.465124E-03 | 5.027782E-05 |
| $\|A(\gamma)\| = \gamma = 0.9$ | | | | |
| $D$ | IT | TS/IT | Average $\|\nabla f(u_{\text{final}})\|$ | Average $f(u_{\text{final}})$ |
| 1 | 548 | 91.24 | 1.584950E-01 | 4.241818E-01 |
| 5 | 488 | 102.45 | 6.499064E-02 | 2.452093E-02 |
| 10 | 479 | 104.38 | 1.062564E-01 | 4.031413E-02 |

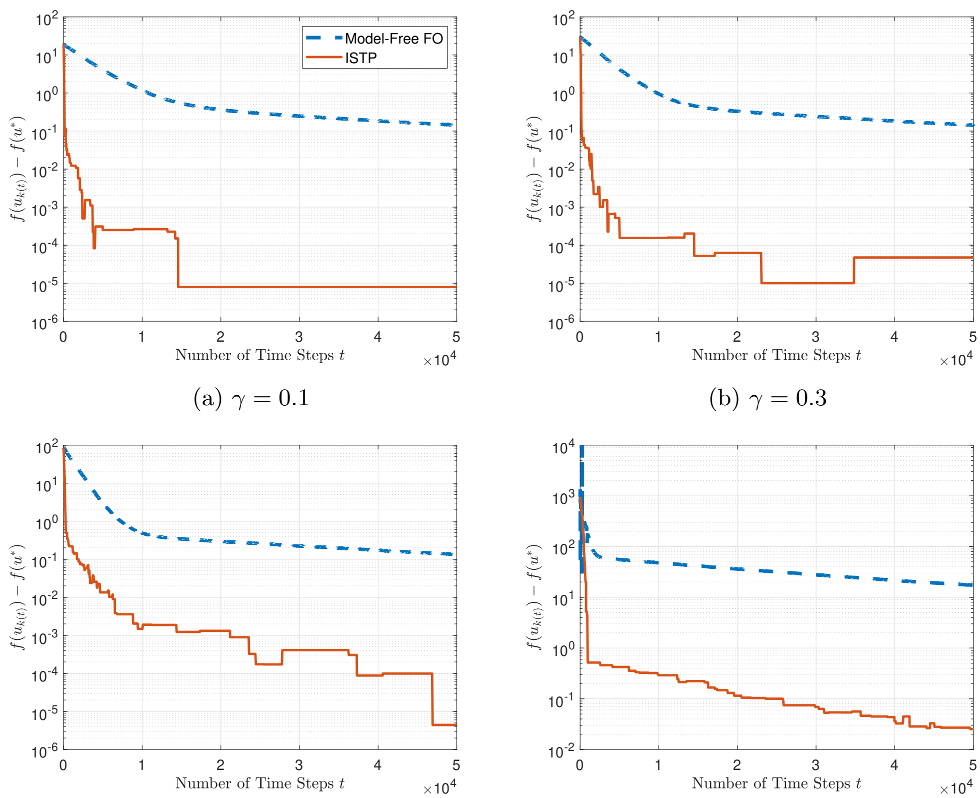
To conclude, we compare ISTP with the model-free feedback optimization method proposed in [21], applied to problem (40) with $\lambda = 0$. In this case, the Model-Free FO method is defined by

$$
\begin{aligned}
x_{k+1} &= Ax_k + Bu_k + d, \\
\Phi_k &= H_\mu(x_{k+1} - \bar{x}), \\
g_k &= \frac{v_k}{\delta}\left(\Phi_k - \Phi_{k-1}\right), \\
w_{k+1} &= w_k - \eta g_k, \\
u_{k+1} &= w_{k+1} + \delta v_{k+1}, \quad v_{k+1} \sim \mathcal{N}(0, I),
\end{aligned}
$$

with initialization $u_0 = w_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$, $x_0 = (I - A)^{-1}d$, $v_0 \sim \mathcal{N}(0, I)$, and $\Phi_{-1} = H_\mu(x_0 - \bar{x})$. We use $\delta = 10^{-2}$ and $\eta = 10^{-3}$. For a fair comparison between the two methods, we consider the sequence $\{u_{k(t)}\}_{t \geq 0}$, where

$$
k(t) = \max\left\{ k \ : \ \sum_{j=0}^{k} t_j \leq t \right\},
$$

and $t_j$ denotes the number of time steps required by iteration $j$. In other words, $k(t)$ is the largest iteration index such that the control $u_{k(t)}$ can be computed within a budget of at most $t$ time steps. For the model-free method, each iteration requires one time step, so $u_{k(t)}$ coincides with $u_t$. Figure 2 reports the evolution of the functional residual $f(u_{k(t)}) - f(u^*)$ as a function of the number of time steps $t$. Since $\lambda = 0$, we have $u^* = \bar{u} = \begin{bmatrix} 10 & 0 & 10 & 0 & 10 \end{bmatrix}^T$ and $f(u^*) = 0$.




(a) $\gamma = 0.1$

(b) $\gamma = 0.3$

(c) $\gamma = 0.6$

(d) $\gamma = 0.9$

Figure 2: Comparison between ISTP and the Model-Free First-Order method from [21].

# 5 Conclusion

In this paper, we proposed the ISTP method, an adaptation of the STP method in [2] to unconstrained minimization problems that relies only on a partially controllable, inexact zeroth-order oracle of the objective function. Assuming that the objective function has Lipschitz continuous gradient, we showed that ISTP takes at most $\mathcal{O}(n|\ln(\epsilon)|^2\epsilon^{-2})$ calls to the inexact oracle to find a point where the expected norm of the gradient is below $\epsilon > 0$. As an application, we considered the use of the new method to solve model-free steady-state optimization problems. Under natural assumptions on the cost function, we proved that for a contracting discrete dynamical system with constant control input $u$, once the distance between two consecutive states is smaller than a given threshold $\delta$, the cost evaluated at the most recent state provides an approximate value $f_\delta(u)$ such that $|f_\delta(u) - f(u)| \leq C\delta$ for some constant $C$. Since this procedure only requires the observation of the states of the system over a certain number of time steps, it allows the use of ISTP for model-free steady-state optimization. We also proved that, in this setting, ISTP takes at most $\tilde{\mathcal{O}}(n\epsilon^{-2})$ time steps of the dynamical system to find a control input whose expected norm of the gradient is below $\epsilon$. Preliminary numerical experiments with a linear dynamical system illustrate our theoretical findings and show that ISTP can be competitive with the model-free feedback optimization method from [21].

As a topic for future research, it would be interesting to generalize ISTP to constrained optimization problems [25] and to investigate its application to bilevel optimization problems [15].

## Code Availability

The MATLAB implementation of the ISTP algorithm used in this work is publicly available at: `https://github.com/geovanigrapiglia/ISTP`.

## Acknowledgements

The author is grateful to Gianluca Bianchin and Amir Mehrnoosh for insightful discussions on model-free steady-state optimization.

## References

[1] A.S. Berahas, R.H. Byrd, J. Nocedal. *Derivative-free optimization of noisy functions via quasi-Newton methods*. SIAM Journal on Optimization 29 (2019), pp. 965–993.

[2] Bergou, E.H., Gorbunov, E., Richtárik, P.: Stochastic Three Points Method for Unconstrained Smooth Minimization. SIAM Journal on Optimization 30, 2726–2749 (2020)

[3] C. Audet and W. Hare, *Derivative-Free and Blackbox Optimization*, Springer, 2017.

[4] S. Boucherouite, G. Malinovsky, P. Richtárik, E.H. Bergou, *Minibatch Stochastic Three Points Method for Unconstrained Smooth Minimization*. Proceedings of the AAAI Conference on Artificial Intelligence 38, 20244–20352 (2024)

[5] C. Audet, S. Le Digabel, V. Rochon Montplaisir, and C. Tribes, *NOMAD version 4: Nonlinear optimization with the MADS algorithm*, ACM Trans. Math. Softw., 48 (2021), pp. 1–22.

[6] W. Wang, Z. He, G. Belgioioso, S. Bolognani, F. Dörfler, *Online Feedback Optimization over Networks: A distributed model-free approach.* IEEE 63rd Conference on Decision and Control (CDC), 2403–2408 (2024)

[7] C. Cartis, L. Roberts. *Scalable subspace methods for derivative-free nonlinear least-squares optimization.* SIAM Journal on Optimization 31 (2021), pp. 3071–3101.

[8] E. Cesaroni, G. Liuzzi, S. Lucidi, *Derivative-free bilevel optimization with inexact lower-level solutions.* ArXiv:2603.20759 (2026)

[9] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, C.-J. Hsieh. *ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models.* Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security (2017), pp. 15–26.

[10] A. R. Conn, K. Scheinberg, and L. N. Vicente, *Introduction to derivative-free optimization*, SIAM, 2009.

[11] A. R. Conn, K. Scheinberg, L. N. Vicente. *Global convergence of general derivative-free trust-region algorithms to first- and second-order critical points.* SIAM Journal on Optimization 20 (2009), pp. 387–415.

[12] A. L. Custódio, J. E. Dennis, Jr., L. N. Vicente. *Using simplex gradients of nonsmooth functions in direct search methods.* IMA Journal of Numerical Analysis 28 (2008), pp. 770–784.

[13] K.J. Dzahini, F. Rinaldi, C.W. Royer, D. Zeffiro, *Direct-search methods in the year 2025: Theoretical guarantees and algorithmic paradigms.* EURO Journal on Computational Optimization 13 (2025), 100110.

[14] K. J. Dzahini, S. M. Wild. *Stochastic trust-region algorithm in random subspaces with convergence and expected complexity analyses.* SIAM Journal on Optimization 34 (2024), pp. 2671–2699.

[15] M.J. Ehrhardt, Lindon Roberts. *Inexact Derivative-Free Optimization for Bilevel Learning.* Journal of Mathematical Imaging and Vision 63 (2021), pp. 580–600.

[16] F. D. Fiore, L. Mainini, *Physics-aware multifidelity Bayesian optimization: A generalized formulation.* Computers & Structures 322 (2024), 107302.

[17] G. François, B. Srinivasan, D. Bonvin, *Comparison of six implicit real-time optimization schems.* Journal Européen des Systemes Automatisés 46 (2012), 291–305.

[18] R. Garmanjani, D. Júdice, and L. N. Vicente, *Trust-region methods without using derivatives: worst case complexity and the nonsmooth case*, SIAM J. Optim., 26 (2016), pp. 1987–2011.

[19] G. N. Grapiglia, *Quadratic regularization methods with finite-difference gradient approximations*, Comput. Optim. Appl., 85 (2023), pp. 683–703.

[20] S. Gratton, C. W. Royer, L. N. Vicente, Z. Zhang. *Direct search based on probabilistic descent.* SIAM Journal on Optimization 25 (2015), pp. 1515–1541.

[21] Z. He, S. Bolognani, J. He, F. Dörfler, X. Guan, *Model-Free Nonlinear Feedback Optimization.* IEEE Transactions on Automatic Control, 69 (2024), pp. 4554–4569.

[22] R. Hooke, T. A. Jeeves. *"Direct search" solution of numerical and statistical problems.* Journal of the ACM 8 (1961), pp. 212–229.

[23] P.D. Khanh, B.S. Mordukhovich, D. B. Tran, *Globally convergent derivative-free methods in nonconvex optimization with and without noise.* Mathematical Programming (2025). DOI: 10.1007/s10107-025-02255-8.

[24] D. Krishnamoorthy, S. Skogestad, *Real-Time optimization as feedback control problem - A review.* Computers and Chemical Engineering 161 (2022), 107723.

[25] D. Krishnamoorthy, F.J. Doyle III, *Model-free real-time optimization of process system using safe Bayesian optimization.* Aiche Jornal 69 (2023)

[26] J. Larson, M. Menickelly, and S. M. Wild, *Derivative-free optimization methods*, Acta Numer., 28 (2019), pp. 287–404.

[27] A. Mehrnoosh, G. Bianchin. *Two-point random gradient-free methods for model-free feedback optimization.* arXiv (2025).

[28] Y. E. Nesterov, V. G. Spokoiny. *Random gradient-free minimization of convex functions.* Foundations of Computational Mathematics 17 (2017), pp. 527–566.

[29] P. Plaban, P. Bachman, A. Renganathan, *Derivative-free optimization is competitive for aerodynamic design optimization in moderate dimensions.* arXiv:2505.09088 (2025).

[30] M. J. Powell, *The NEWUOA software for unconstrained optimization without derivatives*, Large-Scale Nonlinear Optim., (2006), pp. 255–297.

[31] H.-J. M. Shi, M. Q. Xuan, F. Oztoprak, J. Nocedal. *On the numerical performance of finite-difference-based methods for derivative-free optimization.* Optimization Methods and Software 38 (2023), pp. 289–311.

[32] J. Snoek, H. Larochelle, R.P. Adams, *Practical Bayesian Optimization of Machine Learning Algorithms.* NeurIPS 25 (2012), pp. 2960–2968.

[33] S. Tkachenko, A. Andreev, A. Beznosikov, *Three-point method with zero-order oracle that inexactly and randomly compares function values.* International Conference on Computational Optimization, 2024.

[34] V. Torczon. *On the convergence of pattern search algorithms.* SIAM Journal on Optimization 7 (1997), pp. 1–25.

[35] F. Widmer, S.V. Dooren, C.H. Onder, *Optimization of the energy-comfort trade-off of HVAC systems in electric city buses based on a steady-state model.* Control Engineering Practice 154 (2025), 106158.

[36] Z. Zhang, *PRIMA: Reference Implementation for Powell's Methods with Modernization and Amelioration*, DOI: 10.5281/zenodo.8052654, 2023.