Extremum Seeking Algorithms based on Non-Commutative Maps

Jan Feiling ***, Christophe Labar ***,
Victoria Grushkovskaya ****, Emanuele Garone ***,
Michel Kinnaert ****, Christian Ebenbauer *

* Institute for Systems Theory and Automatic Control, University of Stuttgart, Germany (jan.feiling@ist.uni-stuttgart.de)
** Dr. Ing. h.c. F. Porsche AG, Stuttgart, Germany (jan.feiling@porsche.de)
*** Department of Control Engineering and System Analysis, Université libre de Bruxelles, Belgium (chlabar,egarone,michel.kinnaert}@ulb.ac.be)
**** Institute for Mathematics, Julius-Maximilians-University Würzburg, Germany (viktoria.grushkovska@mathematik.uni-wuerzburg.de)

Abstract: In this work, discrete-time extremum seeking algorithms for unconstrained optimization problems are developed. A general class of non-commutative maps and one- and two point function evaluation polices are presented to approximate a gradient-descent algorithm, suitable for extremum seeking problems. Moreover, adaptive step size rules are discussed to achieve faster convergence and vanishing steady state oscillations.

© 2019, IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd. All rights reserved.

Keywords: Extremum Seeking, Optimization, Adaptive Control, Discrete-Time Systems

1. INTRODUCTION

Extremum seeking (ES) is a powerful class of model-free and real-time optimization methods that drive the system towards a steady state that is a (local) optimizer of the output map of the system by utilizing time-periodic signals and output measurements. The first paper of ES probably appears in the early 1920s by Leblanc (1922). Since then many ES approaches (see e.g. Ariyur and Krstić (2003), Guay and Zhang (2003), Atta et al. (2015), Benosman (2017), and Tan et al. (2010) for a detailed survey) have been in the literature, especially with the regained interest initiated by first rigorous proofs of stability for nonlinear continuous-time dynamical systems, based on averaging and singular perturbations (Krstić and Wang (2000); Tan et al. (2010)). ES for discrete-time systems and data-sampled based ES has been considered for example in Choi et al. (2002), Stanković and Stipanović (2009), Teel and Popovic (2001), Khong et al. (2013), and Poveda and Teel (2017). In recent years, a novel class of continuous-time ES schemes has been designed by utilizing non-commutative vector fields and Lie bracket approximation methods. Based on these methods, a general framework to approximate gradients and to design ES for unconstrained, constrained, and distributed optimization and adaptive control problems has been developed, see, e.g., Diür et al. (2013, 2014), Grushkovskaya et al. (2018), Schenk Krisi (2014), Michalowsky et al. (2017), and Labar et al. (2018).

Lie bracket based ES methods are formulated in a continuous-time framework. However, in many applications discrete-time formulations are preferred. Hence, the goal of this work is to design novel discrete-time ES algorithms utilizing non-commutative maps instead of non-commutative vector fields. Leveraging such non-commutative maps for a derivative-free optimization algorithm was first presented in Feiling et al. (2018). The key idea is to use non-commutative maps to evaluate the objective function at certain points such that the composition of the maps approximates a gradient descent step. The difference between derivative-free optimization algorithms and real-time optimization schemes like ES is that only points along the output trajectory of the system can be utilized for optimization in the latter case. The algorithms in Feiling et al. (2018) do not satisfy this condition. Therefore we aim in this paper to modify and extend the algorithms and results in Feiling et al. (2018) such that they can be applied to ES problems.

In particular, the main contribution of this paper are discrete-time ES algorithms for unconstrained optimization problems. We provide a general class of non-commutative maps for approximating a gradient-descent direction based on single-point evaluations along the output trajectory. Moreover, we also present a novel two-point evaluation policy which recovers, for a vanishing step size, the results of the continuous-time ES version based on Lie brackets, cf. Diür et al. (2013). Furthermore, this work discusses some practical tuning rules like adaptive step sizes, such that the convergence is accelerated and a vanishing steady state amplitude is achieved.

Notation. \( \mathbb{R} \) denotes the real and \( \mathbb{N} \) the natural numbers.

The interval of a real number is denoted by \( (a, b) := \{ x \in \mathbb{R} : a < x < b \} \). A \( \delta \)-neighborhood of a point \( x^* \in \mathbb{R} \) with some \( \delta \in (0, \infty) \) is defined as \( \mathbb{B}_\delta(x^*) := \{ x \in \mathbb{R} : |x-x^*| \leq \delta \} \). The Lie bracket between two functions \( f, g : \mathbb{R} \rightarrow \mathbb{R} \) of class \( C^1(\mathbb{R}, \mathbb{R}) \) is denoted by \( [f,g] = \frac{\partial^2}{\partial x \partial y} f - \frac{\partial^2}{\partial y \partial x} g \).
2. PROBLEM STATEMENT

As indicated in the introduction, this work aims to tackle ES problems related to unconstrained minimization

$$\min_{x \in \mathbb{R}} J(x),$$

where no analytical expression of $J(x)$ is available. Throughout the paper we assume $J \in C^2(\mathbb{R}, \mathbb{R})$, $J(x)$ is radially unbounded and there exists a $x^* \in \mathbb{R}$ such that $\nabla J(x)(x - x^*) > 0$ for all $x \in \mathbb{R}\{x^*\}$ and refer to it as Assumption 1. For the sake of simplicity we consider static systems $y = J(x)$, where $y$ is the system output and $x$ is a scalar input. However, the ideas of this paper can be converted to more general problems including dynamical systems, constraint optimization or distributed ES problems.

In this work we develop algorithms of the form

$$x_{k+1} = M^\overline{\nabla}M(x_k, x_{k-1}, \ldots), \quad k \geq 0$$

(2)

with transition maps $M^\overline{\nabla}_k(\cdot)$ and step size $h \in (0, \infty)$. To address problems as in (1) in an ES fashion we impose the following conditions:

(C1) At time step $k$, each iteration $M^\overline{\nabla}_k(\cdot)$ utilizes solely points $x_i$ and objective evaluations $y_i = J(x_i)$ with $0 \leq i \leq k$, meaning that only past information along the trajectory of the algorithm can be used.

(C2) For every $k \in \mathbb{N}$ the composition of $n \in \mathbb{N}$ transition maps

$$x_{k+m} = \left(M^\overline{\nabla}_{k+m-1} \circ \cdots \circ M^\overline{\nabla}_k\right)(x_k, x_{k-1}, \ldots)$$

(3)

has to approximate a gradient-descent step, i.e.,

$$x_{k+m} = x_k - h\alpha \nabla J(x_k) + O(h^{3/2})$$

(4)

where $\alpha \in (0, \infty)$ is some scaling factor.

A partial solution to the problem stated above was presented by the discrete-time derivative-free optimization algorithms in Feiling et al. (2018). The transition maps are given by a single-point approach

$$M^\overline{\nabla}_k(x_k) = E^\overline{\nabla}_k(x_k) := x_k + \sqrt{h} s_k(J(x_k))$$

(5)

and a two-point approach

$$M^\overline{\nabla}_k(x_k) = H^\overline{\nabla}_k(x_k) := x_k + \sqrt{h/2} \left[ s_k(J(x_k)) + hJ(x_k) + \sqrt{h} s_k(J(x_k)) \right]$$

(6)

with so-called evaluation maps

$$s_t(J(x_k)) = f(J(x_k))u_t + g(J(x_k))v_t,$$

(7)

defined by functions $f, g : \mathbb{R} \to \mathbb{R}$ and exploration sequences $u_t, v_t \in \mathbb{R}$. The two approaches (5) and (6) are called single- and two-point approaches, respectively, since the number of function evaluations of $J(x)$ in each iteration is one and two, respectively.

Condition (C1) is satisfied by (5), but not by (6), due to the necessary evaluation of $J$ at the point $x_k + \sqrt{h} s_k(J(x_k))$. It can be shown that (C2) is satisfied by (5) with $f(J(x)) = \sin(J(x))$, $g(J(x)) = \cos(J(x))$, $m = 4$, and

$$u_k = \begin{cases} 1 & k = 0 \\ 0 & k = 1 \\ -1 & k = 2 \\ 0 & k = 3 \\ \text{else} & u_{k-4} \end{cases}, \quad v_k = \begin{cases} 0 & k = 0 \\ 0 & k = 2 \\ -1 & k = 3 \\ \text{else} & v_{k-4} \end{cases}$$

(8)

as presented in Feiling et al. (2018).

In the following we extend these results and adapt them to ES problems. First we derive a general class of functions $f$ and $g$ for the single-point approach (5) such that (C2) holds. Secondly, a novel two-point approach satisfying (C1) and (C2) is proposed. Finally, we discuss an adaptive step size rule $h_{k+1} = \mu h_k$ with $\mu > 0$ depending on some approximation error in order to accelerate convergence and to achieve a vanishing steady state amplitude behavior.

3. MAIN RESULTS

3.1 Single-Point Approach

In this section we consider the single-point algorithm based on transition map (5)

$$x_{k+1} = E^\overline{\nabla}_k(x_k) = x_k + \sqrt{h} s_k(J(x_k))$$

(9)

with $s_t(J(x_k))$ as in (7) and $u_k, v_k$ as in (8).

The evolution of $x_{k+4}$ w.r.t. $x_k$ and $k \in \mathbb{N}$, induced by the non-commutative transition map (5) is stated in the following (Feiling et al., 2018, Lemma 1):

**Lemma 1.** Suppose that Assumption 1 holds. Let $f, g \in C^2(\mathbb{R}, \mathbb{R})$ hold. Consider algorithm (9) and assume that $x_t \in \mathcal{V}$ for all $\ell = k, k+1, k+2, k+3$, where $\mathcal{V} \subset \mathbb{R}$ is some bounded convex set. Then it holds

$$x_{k+4} = x_k + O(h^{3/2})$$

(10)

By comparing the expression in the curly brackets of (10) with the continuous-time ES version based on Lie brackets, cf. D" urr et al. (2013), an additional term $\frac{df^2 + g^2}{2} \partial J$ of $O(h)$ is present. Hence, for $h \to 0$ the continuous-time ES case based on Lie brackets is not recovered anymore. Therewith the class of functions $f, g$ presented in Grushkovskaya et al. (2018), such that $[f, g](z) = \alpha(z)$ with some user-defined function $\alpha : \mathbb{R} \to \mathbb{R}$, in order to approximate a gradient-descent algorithm and satisfying (C2), is not applicable anymore. Our first main result of this paper characterizes a general class of functions $f$ and $g$ that satisfies (C2) for algorithm (9), and such that the term in the curly bracket of (10) is some user-defined function.

**Theorem 1.** Suppose that Assumption 1 holds. Let $r, \alpha : \mathbb{R} \to (0, \infty)$ be arbitrary and define $\phi : \mathbb{R} \to \mathbb{R}$ as

$$\phi(z) = \frac{1}{2} \ln(r(z)) - \int \alpha(z) dz + c, \quad c \in \mathbb{R}. \quad (11)$$

Then by selecting $f, g \in C^2(\mathbb{R}, \mathbb{R})$ as

$$f(z) = \sqrt{r(z)} \cos(\phi(z)), \quad g(z) = \sqrt{r(z)} \sin(\phi(z))$$

(12)

evolution of the algorithm (9) is given by

$$x_{k+4} = x_k - h\alpha(J(x_k))\nabla J(x_k) + O(h^{3/2})$$

(13)

for all $k \in \mathbb{N}$.

**Proof.** Let $f(z)$ and $g(z)$ be given by (12) and take $z = J(x)$. Direct computations show that, on the one hand, the function $\phi(z)$ given by (11) with any $r$ and $\alpha$ satisfies the identity

$$r(J(x)) \frac{\partial \phi}{\partial J}(J(x)) - \frac{1}{2} r(J(x)) \frac{\partial r}{\partial J}(J(x)) = -\alpha(J(x)). \quad (14)$$
On the other hand, if the functions \( f(J(x)) \) and \( g(J(x)) \) are given by (12) with (11), then
\[
[f, g](J(x_k)) - \frac{1}{2} \frac{\partial f^2 + g^2}{\partial J}(J(x_k)) = \frac{1}{2} \frac{\partial r}{\partial J}(J(x_k)) - \frac{1}{2} \frac{\partial r}{\partial J}(J(x_k)).
\]  
(15)
Substituting (14) and (15) in (10) we get (13). 

**Remark 1.** Note that for \( \alpha(J(x)) = 1 \) a gradient-like step
\[
x_{k+1} = x_k - h \nabla J(x_k) + O(h^{3/2})
\]  
(16)
is approximated by algorithm (9) incorporated with the class of functions given in (12) with (11). As possible examples, with \( f(z) \) and \( g(z) \) satisfying (12) and (11), one may consider:
\[
\begin{align*}
  f(z) &= \cos(z), \\
  g(z) &= -\sin(z); \\
  f(z) &= \sqrt{z} \cos(\ln(\sqrt{z})), \\
  g(z) &= -\sqrt{z} \sin(\ln(\sqrt{z})); \\
  f(z) &= \sqrt{z} \cos(\phi_1), \\
  g(z) &= \sqrt{z} \sin(\phi_1); \\
  f(z) &= \sqrt{z}^2 \cos(\phi_2), \\
  g(z) &= \sqrt{z}^2 \sin(\phi_2);
\end{align*}
\]
(17) (18) (19) (20)
where \( \phi_1 = \frac{t}{2} z + e^{-z} \) and \( \phi_2 = \frac{t}{2} \ln(z) + \frac{i}{t} z^{-1} \) with \( t \in (1, \infty) \). Note that to ensure \( z > 0 \) in (18) and (20), some weak prior knowledge about the objective \( J(x) \) is necessary. Moreover, if \( z^* = J(x^*) \) approaches zero, the steady state oscillation also tends to zero for (18) and (20).

Based on Lemma 1 and Theorem 1 semi-global practical asymptotic convergence to \( x^* \) can be established. 

**Theorem 2.** Let Assumption 1 hold. Consider functions \( f(J(x)) \) and \( g(J(x)) \) as in (12) with (11) and \( \alpha(J(x)) = 1 \). Then, for all \( \delta_1, \delta_2 \in (0, \infty) \) with \( \delta_2 < \delta_1 \), there exist an \( h^* \in (0, \infty) \) and \( N \in \mathbb{N} \), such that for all \( h \in (0, h^*) \) and \( x_0 \in \mathcal{U}^2_{\delta_1} \), it holds \( x_k \in \mathcal{U}^2_{\delta_2} \) for all \( k \geq N \).

The proof of Theorem 2 follows along the lines of the proof of (Feiling et al., 2018, Theorem 2) by utilizing Lemma 1 and Theorem 1.

### 3.2 Two-Point Approach

As discussed in Section 2 and Section 3.1, the approach (6) does not fulfill condition (C1) and hence cannot be applied to ES problems. The reason for this is that the point \( x_k + \sqrt{h} s_h(x_k) \) is not a point along the trajectory of the algorithm, but the value at this point is needed to compute \( x_{k+1} \). On the other hand the single-point approach does not recover the continuous-time ES version based on Lie brackets in the limit \( h \to 0 \), cf. Dürr et al. (2013). In order to fix this problem and get a scheme which approximates the continuous-time ES case based on Lie brackets, a novel two-point approach is presented, which satisfies conditions (C1) and (C2), as stated in the following:
\[
x_{k+1} = C_k \sqrt{h} [x_k, x_{k-1}, x_{k-2}]
\]
:= \sqrt{h} \left[ s_{k-2}(J(x_k)) + s_{k-1}(J(x_k-1)) \right]
\]
with \( s_{\ell}(J(x_k)) \) as in (7) and
\[
u_k = \begin{cases} 
1 & k = 0 \\
0 & k = 1 \\
-1 & k = 2 \\
0 & k = 3 \\
0 & k = 4 \\
0 & \text{else}
\end{cases}
\]
(21)
Note that \( v_k \) in (21) has different signs compared to (8). As in Lemma 1, an analog result for the evolution of \( x_{k+4} \) w.r.t. \( x_k \) and \( k \in \mathbb{N} \) is gained for algorithm (21) as presented in the following theorem:

**Theorem 3.** Suppose that Assumption 1 and \( f, g \in C^2(\mathbb{R}, \mathbb{R}) \) hold. Consider algorithm (21) and assume that \( x_k \in \mathcal{V} \) for all \( \ell = k - 4, k - 3, \ldots, k + 2 \), where \( \mathcal{V} \) is some bounded convex set. Then it holds
\[
x_{k+4} = x_k + h[f, g](J(x_k)) \nabla J(x_k) + O(h^{3/2}).
\]  
(22)
The proof of Theorem 3 is given in Appendix A.

**Remark 2.** For the limit case \( h \to 0 \) the second order approximation (22) of algorithm (21) approaches the continuous-time ES scheme based on Lie-brackets, cf. Dürr et al. (2013).

**Remark 3.** Since the term of \( O(h) \) in (22) is the Lie bracket, we can use the class of functions \( f(z) \) and \( g(z) \) with \( z = J(x) \) presented in Grushkovskaya et al. (2018) and given by
\[
g(z) = -f(z) \int \frac{\alpha(z)}{f(z)^2} dz.
\]  
(23)
such that \( [f, g](z) = \alpha(z) \). Then (C2) is obtained with \( \alpha(z) = \tilde{\alpha} \) with \( \tilde{\alpha} \in (0, \infty) \).

**Remark 4.** If we consider the two-point approach (21) as a numerical integration method for \( \dot{x} = f(J(x))u + g(J(x))v \), similar to the second order Heun (trapezoidal) method, one has the same second order approximation result, cf. (Feiling et al., 2018, Lemma 2). In contrast to the Heun method, however, one needs less function evaluations, due to utilizing each point in two consecutive iterations, hence only one evaluation map \( s_{\ell}(J(x_k)) \) has to be evaluated at each iteration.

Based on Theorem 3, semi-global practical asymptotic convergence to \( x^* \), similar to Theorem 2, can be established.

**Theorem 4.** Let Assumption 1 hold. Consider functions \( f(J(x)) \) and \( g(J(x)) \) as in (23) with \( \alpha(J(x)) = 1 \). Then, for all \( \delta_1, \delta_2 \in (0, \infty) \) with \( \delta_2 < \delta_1 \), there exist an \( h^* \in (0, \infty) \) and \( N \in \mathbb{N} \), such that for all \( h \in (0, h^*) \) and \( x_0 \in \mathcal{U}^2_{\delta_1} \), it holds \( x_k \in \mathcal{U}^2_{\delta_2} \) for all \( k \geq N \).

The proof of Theorem 4 follows the lines of the proof of (Feiling et al., 2018, Theorem 2) by utilizing Theorem 3.

### 3.3 Adaptive Step Size Rule

The presented algorithms (9) and (21), including (5) and (6), suffer in convergence speed for small step sizes and show large oscillations in a neighborhood of convergence for large step sizes (depending on choice of functions \( f(J(x)) \) and \( g(J(x)) \) and objective \( J(x) \)). From a practical and implementation perspective it is straightforward to investigate adaptive step sizes to achieve faster convergence plus a good steady state behavior, i.e., a small neighborhood of convergence around a local optimizer. In the following, those points are addressed.

**Accelerated Convergence.** Consider the single- and two-point approaches (9) and (21). Let us execute one two-point update \( x_{k+1} \) and in parallel a single-point update \( x_{k+1} \), both w.r.t. \( x_k \). Then the difference of the map approximations is
\( e_k = x_{k+1} - \hat{x}_{k+1} \)
\[ = \frac{\sqrt{2}}{2} [s_{k-2}(J(x_{k-2})) + s_{k-1}(J(x_{k-1})) - 2\hat{s}_k(J(x_k))] \] (24)
where \( \hat{s}_k(J(x_k)) \) belongs to the single-point algorithm (9). Given error thresholds \( \epsilon_1, \epsilon_2 \in (0, \infty) \) with \( \epsilon_1 > \epsilon_2 \) and multiplication factors \( \mu_1 \in (1, \infty) \) and \( \mu_2 \in (0, 1) \), an adaptive step size can be implemented, similar to the well known adaptive Runge-Kutta procedures (Cheney and Kincaid, 2012, Chapter 7.3) as follows
\[
h_{k+1} = \begin{cases} 
\mu_1 h_k & \text{if } |e_k| \leq \epsilon_1 \\
\mu h_k & \text{if } \epsilon_1 < |e_k| < \epsilon_2 \\
\mu_2 h_k & \text{if } \epsilon_2 \leq |e_k|
\end{cases}
\] (25)

This step size update rule accelerates convergence (see e.g. Section 4), but on the other hand, it implicates a larger neighborhood of convergence which is of order \( O(\sqrt{h_k}) \) based on error thresholds \( \epsilon_1 \) and \( \epsilon_2 \).

**Vanishing Steady State Amplitude.** Error thresholds \( \epsilon_1 \) and \( \epsilon_2 \) in (25) can be reduced, which is equivalent to reducing the steady state amplitude, as soon as a neighborhood of convergence is detected. Hence, if \( |x_{k+4} - x_k| \leq \epsilon_3 \sqrt{h_k} \) for some error threshold \( \epsilon_3 \in (0, \infty) \), then with a \( \nu \in (0, 1) \) apply
\[ \epsilon_1 = \nu \epsilon_1, \quad \epsilon_2 = \nu \epsilon_2. \] (26)

Combining (25) and (26) yields to faster convergence and vanishing steady state amplitude (see e.g. Section 4). However, in the case of a dynamical optimizer, e.g. set point changes, a procedure to increase error thresholds \( \epsilon_1 \) and \( \epsilon_2 \) to accelerate convergence again is required, but not considered in this work.

Summarizing, the adaptive single- and two-point ES algorithms based on non-commutative maps with adaptive step size is presented in Algorithm 1.

**Algorithm 1 ES Algorithm based on non-commutative maps with adaptive step size**

1. **Required:** \( x_0, h, \epsilon_1, \epsilon_2, \epsilon_3, \mu_1, \mu_2, \nu, \) stop criterion \( f(J(x)), g(J(x)) \) as in (12)
2. \( s_t(J(x_t)) \) with \( u_t, v_t \) as in (21), \( \hat{s}_t(J(x_t)) \) with \( u_t, v_t \) as in (9)
3. **Init:** \( k = 0; \ x_{-2} = x_{-1} = x_0 \)
4. **while** stop criterion is not fulfilled do
5. \( x_{k+1} = x_k + \frac{\sqrt{2}}{2} [s_{k-2}(J(x_{k-2})) + s_{k-1}(J(x_{k-1}))] \)
6. \( \hat{x}_{k+1} = \hat{x}_k + \sqrt{h_k}(J(x_k)) \)
7. \( e_k \leftarrow x_{k+1} - \hat{x}_{k+1} \)
8. \( h_{k+1} \leftarrow \begin{cases} 
\mu_1 h_k & \text{if } |e_k| \leq \epsilon_1 \\
\mu h_k & \text{if } \epsilon_1 < |e_k| < \epsilon_2 \\
\mu_2 h_k & \text{if } \epsilon_2 \leq |e_k|
\end{cases} \)
9. **if** \( |x_{k+4} - x_k| \leq \epsilon_3 \sqrt{h_k} \) **then**
10. \( \epsilon_1 \leftarrow \nu \epsilon_1 \)
11. \( \epsilon_2 \leftarrow \nu \epsilon_2 \)
12. **end if**
13. \( k \leftarrow k + 1 \)
14. **end while**
15. **return** \( [x_0, x_1, \ldots] \)

4. **NUMERICAL EXAMPLES**

In this section we study numerical examples of the proposed algorithms (9), (21), and Algorithm 1 incorporated with different function pairs \( f(J(x)) \) and \( g(J(x)) \) given by (12) with (11) and (23), respectively, in various simulation examples. To this end, we consider the cost function \( J(x) = (x - 1)^2 \) with minimizer \( x^* = 1 \) and optimal value \( J^* = 0 \). Note that algorithm (21) is initialized with \( x_{-2} = x_{-1} = x_0 \).

In a first simulation we study the class of function given in (12) with (11) incorporated in algorithm (9). Namely, the different trajectories with initial value \( x_0 = 2 \), step size \( h = 0.05 \) and function pairs (17) - (20) are investigated and are depicted in Figure 1.

![Fig. 1. Trajectories of algorithm (9) incorporated with functions \( f(J(x)) \) and \( g(J(x)) \) of the class (12), where (--): (17), (-): (18), (---): (19) and (--) (20) with \( t = 2 \).](image)

One can observe that (18) and (20) tend to a vanishing steady state amplitude, due to \( J^* = 0 \). Note that the knowledge of \( J^* \) is present in many synchronization and calibration tasks. However, (20) behaves sensitive w.r.t. the variable \( t \), where simulation studies reveal that \( t \) should be less or equal to the largest power of the unknown function \( J(x) \). On the other hand, (17) has a constant amplitude of one, reasoned by the bounded functions \( \sin(\cdot) \) and \( \cos(\cdot) \) and are therefore not sensitive w.r.t. large values of \( J(x) \), whereas (18)-(20) show this behavior.

Our second simulation study carries out the investigation how the proposed adaptive step size rule (25) with (26) accelerates convergence and reduces the steady state amplitude. Hence, Algorithm 1 and algorithm (21) with function pair (17) is initialized with \( x_0 = 2 \) and step sizes \( h = \{10^{-3}, 10^{-2}, 10^{-3}\} \). Furthermore, the parameters \( \epsilon_1 = 0.2, \epsilon_2 = 0.4, \epsilon_3 = 0.05, \mu_1 = 1.5, \mu_2 = 0.9 \) and \( \nu = \{0.75, 1\} \) are chosen. The trajectories of the aforementioned algorithms are pictured in Figure 2. As expected, the proposed adaptive step size rule accelerates convergence such that a neighborhood of the optimizer \( x^* = 1 \) is reached in 15-25 steps for the given setting. Interestingly, starting with a small and non-sensitive step size, Algorithm 1 needs
only a few steps to accelerate, whereas algorithm (21) with a fixed step size has a much slower convergence speed, but a small steady state amplitude depending on \( O(\sqrt{h}) \).

Moreover, a vanishing steady state amplitude is gained by iteratively reducing the neighborhood of convergence, hence \( \nu = 0.75 \) is chosen. In contrast, if \( \nu = 1 \), as in the bottom subplot of Figure 2, we observe a large neighborhood of convergence.

Eventually, a two-dimensional example with adaptive step size is studied. Therefore we extend the cost function to \( J(x) = (x - [1, 2]^\top)^2 \) and the two-dimensional exploration sequences \( \bar{x}_k = [x_k, u_k+1]^\top \), \( \bar{v}_k = [v_k, v_k+1]^\top \) are applied to Algorithm 1. In the two dimensional case it is easy to verify, by a Taylor expansion, that \( \bar{x}_k \) and \( \bar{v}_k \) associated with algorithm (21) reveal the same expression as (22). The result of the two-dimensional simulation example, introduced above, with function pairs (17), initial step size \( h = 10^{-4} \), initial value \( x_0 = [0, 3]^\top \), and parameters \( \epsilon_1, \epsilon_2, \epsilon_3, \mu_1, \mu_2 \) and \( \nu \) as used in the second simulation example, is illustrated in Figure 3. One observes an acceleration of convergence in the beginning and an vanishing steady state amplitude similar to the second simulation example.

5. CONCLUSION

In this work, a novel class of discrete-time extremum seeking algorithms based on non-commutative maps is presented. A single- and two-point approach of the algorithm is developed to approximate a gradient-descent algorithm by utilizing solely points along the output trajectory. In particular, for the single-point approach a general class of functions such that the composition of four maps reveals a gradient-descent step approximation is presented. For the two-point approach we developed a novel update rule, hence a map approximation, such that for a vanishing step size, the continuous-time extremum seeking version based on Lie brackets is recovered. For both versions of the algorithm, semi-global practical convergence to a local optimizer is established. Moreover, an adaptive step size rule is introduced which leads to an acceleration and a vanishing steady state amplitude of the discrete-time extremum seeking algorithms. All results are applied in simulation examples and their behavior is analyzed.

Future work will address the extension to the general multi-dimensional case in a more sophisticated way than a simple coordinate wise descent implementation, which was not addressed in this work, solely a simple extension to a two dimensional problem.

REFERENCES

Ariyur, K.B. and Krstić, M. (2003). Real-time optimization by extremum-seeking control. John Wiley & Sons.

Atta, K.T., Johansson, A., and Gustafsson, T. (2015). Extremum seeking control based on phasor estimation. Systems & Control Letters, 85, 37–45.

Benosman, M. (2017). Learning-Based Adaptive Control: An Extremum Seeking Approach - Theory and Applications. Butterworth-Heinemann.

Cheney, E.W. and Kincaid, D.R. (2012). Numerical mathematics and computing. Cengage Learning.

Choi, J.Y., Krstić, M., Ariyur, K.B., and Lee, J.S. (2002). Extremum seeking control for discrete-time systems. IEEE Transactions on automatic control, 47(2), 318–323.

Dürr, H.B., Stanković, M.S., Ebenbauer, C., and Johansson, K.H. (2013). Lie bracket approximation of extremum seeking systems. Automatica, 49(6), 1538–1552.

Dürr, H.B., Stanković, M.S., Johansson, K.H., and Ebenbauer, C. (2014). Extremum seeking on submanifolds in the Euclidian space. Automatica, 50(10), 2591–2596.

Feiling, J., Zeller, A., and Ebenbauer, C. (2018). Derivative-free optimization algorithms based on non-commutative maps. IEEE Control Systems Letters, 2(4), 743–748.
Grushkovskaya, V., Zuyev, A., and Ebenbauer, C. (2018). On a class of generating vector fields for the extremum seeking problem: Lie bracket approximation and stability properties. *Automatica*, 94, 151–160.

Guay, M. and Zhang, T. (2003). Adaptive extremum seeking control of nonlinear dynamic systems with parametric uncertainties. *Automatica*, 39(7), 1283–1293.

Khang, S.Z., Nešić, D., Tan, Y., and Manzie, C. (2013). Unified frameworks for sampled-data extremum seeking control: Global optimisation and multi-unit systems. *Automatica*, 49(9), 2720–2733.

Krstić, M. and Wang, H.H. (2000). Stability of extremum seeking problem: Lie bracket approximation and stability. *Automatica*, 36(4), 595–601.

Labar, C., Feiling, J., and Ebenbauer, C. (2018). Gradient-based extremum seeking: Performance tuning via Lie bracket approximations. In *2018 European Control Conference (ECC)*, 2775–2780. IEEE.

Leblanc, M. (1922). Sur l’électrification des chemins de fer au moyen de courants alternatifs de fréquence elevee. *Revue Générale de l’électricité*, 12(8), 275–277.

Michalowsky, S., Gharesifard, B., and Ebenbauer, C. (2017). Distributed extremum seeking over directed graphs. In *Proc. 56th IEEE Conference on Decision and Control*, 2095–2101.

Poveda, J.I. and Teel, A.R. (2017). A robust event-triggered approach for fast-sampled-data extremum seeking and learning. *IEEE Transactions on Automatic Control*, 62(10), 4949–4964.

Scheinker, A. and Krstić, M. (2014). Extremum seeking with bounded update rates. *Systems & Control Letters*, 63, 25–31.

Stanković, M.S. and Stipanović, D.M. (2009). Discrete time extremum seeking by autonomous vehicles in a stochastic environment. In *48th IEEE Conference on Decision and Control (CDC)*, 4541–4546.

Tan, Y., Moase, W., Manzie, C., Nešić, D., and Mareels, I. (2010). Extremum seeking from 1922 to 2010. In *Proc. 29th Chinese Control Conference*, 14–26.

Teel, A.R. and Popovic, D. (2001). Solving smooth and nonsmooth multivariable extremum seeking problems by the methods of nonlinear programming. In *Proc. American Control Conference*, 2394–2399.

### Appendix A. PROOF OF THEOREM 3

Evaluating $x_{k+1}$ in (21) for $k = 0, 1, 2, 3$, while considering the definition of $u_t$, $v_t$, and $s_t(J(x_t))$ in (21) for $t = -2, -1, 0, 1, 2$, one has

$$x_4 = x_0 + \sqrt{h} \left[ -f(J(x_{-2})) + 2g(J(x_{-1})) + 2f(J(x_0)) - 2g(J(x_1)) - f(J(x_2)) \right].$$

(A.1)

Performing a first order Taylor expansion with Lagrange remainder of $f(J(x_p))$ around $x_p$ for $p = -2, 2$ leads to

$$f(J(x_p)) = f(J(x_0)) + \frac{\partial f}{\partial J}(J(x_0)) \nabla J(x_0)^{\top} (x_p - x_0) + \frac{\partial^2 f}{\partial J^2}(J(x_p), x_p) (x_p - x_0)^2$$

(A.2)

with $x_p, x_0 \in (x_p, x_0)$. Note that the same expression (A.2) can be obtained for $g(J(x_1))$ and $g(J(x_{-1}))$ by replacing $f(\cdot)$ by $g(\cdot)$. Substituting (A.2) in (A.1) results in

$$x_4 = x_0 + \sqrt{h} \left[ \frac{\partial f}{\partial J}(J(x_0)) \nabla J(x_0)^{\top} (2x_0 - x_2 - x_{-2}) + 2 \frac{\partial g}{\partial J}(J(x_0)) \nabla J(x_0)^{\top} (x_{-1} - x_1) \right] + \sqrt{h} R_1,$$

(A.3)

where

$$R_1 = -\frac{1}{2} \frac{\partial^2 f}{\partial J^2}(J(x_{-2})) \frac{\partial^2 J}{\partial x^2}(x_{-2}, 0) (x_{-2} - x_0)^2 + \frac{\partial^2 f}{\partial J^2}(J(x_{-1})) \frac{\partial^2 J}{\partial x^2}(x_{-1}, 0) (x_{-1} - x_0)^2 - \frac{\partial^2 f}{\partial J^2}(J(x_0)) \frac{\partial^2 J}{\partial x^2}(x_0, 1) (x_0 - x_{1})^2 - \frac{1}{2} \frac{\partial^2 f}{\partial J^2}(J(x_2)) \frac{\partial^2 J}{\partial x^2}(x_2, 0) (x_2 - x_0)^2.$$

(A.4)

Since $f, g, J \in C^2(\mathbb{R}, \mathbb{R})$ is assumed and $X_{i,0} \in \mathbb{V}$, since $\mathbb{V}$ is a convex set and $x_i \in \mathbb{V}$ for $i = -2, -1, 0, 1, 2$, the Hessians $\frac{\partial^2 f}{\partial J^2}(J(x_{i,0})) \frac{\partial^2 J}{\partial x^2}(X_{i,0})$ are bounded. Hence $R_1 = O(\sqrt{h})$.

Then substituting $s_l(J(x_l))$ in (A.5) for $l = -2, -1, 1, 2$ with its zero order Taylor expansion around $x_0$ with Lagrange remainder yields

$$x_m - x_0 = \text{sgn}(m) \sqrt{\frac{h}{2}} \left[ \sum_{i = \min\{0, m\}}^{\max\{-1, m-1\}} s_{i-2}(J(x_{i-2})) + s_{i-1}(J(x_{i-1})) \right].$$

(A.5)

Substituting (A.6) in (A.3) with respect to $u_t, v_t$, and $s_t(J(x_t))$ in (21) for $t = -2, -1, 0, 1, 2$ yields

$$x_4 = x_0 + h f(J(x_0)) g(J(x_0)) \nabla J(x_0) + \sqrt{h} R_1 + h R_2 = x_0 + h [f(J(x_0)) g(J(x_0)) \nabla J(x_0) + \sqrt{h} R_1 + h R_2],$$

(A.7)

where

$$R_2 = \frac{\partial f}{\partial J}(J(x_{-1})) \nabla J(x_{-1})^{\top} (x_{-1} - x_0) + \frac{1}{2} \frac{\partial^2 f}{\partial J^2}(J(x_{-4}, 0)) \nabla J(x_{-4})^{\top} (x_{-4} - x_0) - \frac{\partial g}{\partial J}(J(x_{-3})) \nabla J(x_{-3})^{\top} (x_{-3} - x_0) + \frac{\partial g}{\partial J}(J(x_{-2})) \nabla J(x_{-2})^{\top} (x_{-2} - x_0) + 2 \frac{\partial^2 f}{\partial J^2}(J(x_2)) \nabla J(x_{-2})^{\top} (x_{-2} - x_0) - \frac{\partial g}{\partial J}(J(x_{-2})) \nabla J(x_{-2})^{\top} (x_{-2} - x_0) \nabla J(x_0).$$

(A.8)