BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 63, No. 4, 2015
DOI: 10.1515/bpasts-2015-0098

Plug-in direct particle swarm repetitive controller with a reduced dimensionality of a fitness landscape – a multi-swarm approach

B. UFNALSKI[*] and L.M. GRZESIAK

Faculty of Electrical Engineering, Warsaw University of Technology, 75 Koszykowa St., 00-662 Warsaw, Poland

Abstract. The paper describes a modification to the recently developed plug-in direct particle swarm repetitive controller (PDPSRC) for the sine-wave constant-amplitude constant-frequency (CACF) voltage-source inverter (VSI). The original PDPSRC algorithm assumes that the particle swarm optimizer (PSO) takes into account a performance index defined over the whole reference signal period. Each particle stores all the samples of the control signal, e.g. $\alpha = 200$ samples for a controller working at 10 kHz and the reference frequency equal to 50 Hz. Therefore, the fitness landscape (i.e. the performance index) is α-dimensional (αD), which makes optimization challenging. That solution can be categorized as the single-swarm one. It has been previously shown that the swarm controller does not suffer from long-term stability issues encountered in the classic iterative learning controllers (ILC). However, the convergence of the swarm has to be kept at a relatively low rate to enable successful exploitation in the αD search space, which in turn results in slow responsiveness of the PDPSRC. Here a multi-swarm approach is proposed in which we divide a dynamic optimization problem (DOP) among less dimensional swarms. The reference signal period is segmented into shorter intervals and the control signal is optimized in each interval independently by separate swarms. The effectiveness of the proposed approach is illustrated with the help of numerical experiments on the CACF VSI with an output *LC* filter operating under nonlinear loads.

Key words: repetitive process control, dynamic optimization problem, particle swarm optimizer, repetitive disturbance rejection, non-interacting subswarms, dimension-reduced fitness functional.

Nomenclature

2D	–	two-dimensional (here control system),
αD	–	α-dimensional (here optimization problem),
CACF	–	Constant-Amplitude Constant-Frequency (converter),
DFF	–	Disturbance Feed-Forward,
DOP	–	Dynamic Optimization Problem,
FSF	–	Full-State Feedback (controller),
ILC	–	Iterative Learning Control(ler),
k-direction	–	pass-to-pass direction,
MMO	–	Multi-Modal Optimization,
p-direction	–	along the pass direction,
PDPSRC	–	Plug-in Direct Particle Swarm Repetitive Controller (a basic approach),
PDMSRC	–	Plug-in Direct Multi-Swarm Repetitive Controller (a novel approach),
PSO	–	Particle Swarm Optimization (-er),
RC	–	Repetitive Control(ler),
RFF	–	Reference Feed-Forward,
RMSE	–	Root Mean Squared Error (here calculated within one period of a reference signal),
VSI	–	Voltage-Source Inverter,
α	–	number of samples per reference signal period,
α_n	–	points of swarm division (points of subswarms' adjacency),
i	–	swarm iteration identification number/index,
j	–	particle identification number/index,
$\mathcal{J}$	–	cost functional,
k	–	pass (reference signal period) number,
n	–	subswarm identification number/index,
p	–	sample identification number,
$\bullet^{\mathrm{m}}$	–	measurement signal corrupted with noise,
$\bullet^{\mathrm{ref}}$	–	reference signal.

1. Introduction

The PDPSRC [1] was initially proposed for CACF inverters with an LC output filter but its use is not limited to this kind of power electronic converters. That was the first step towards a novel versatile stochastic repetitive controller. Its development was motivated by unsatisfactory results obtained using the classic ILC scheme. Most of the iterative learning controllers (ILC) suffer from long term stability problems and additional filtering is essential to stabilize the system [2–6]. For example, the very basic P-type control law has to be modified into

$$u(p,k) = \mathbf{Q}(z^{-1})u(p,k-1) + \mathbf{L}(z^{-1})k_{\mathrm{RC}}e(p,k-1), \quad (1)$$

where u denotes the control signal, e is the control error, k_{RC} is the controller gain, k is the iteration (pass, trial, cycle) index, p is the time index along the pass ($1 \leq p \leq \alpha$, where α is the pass length), with $\mathbf{Q}$ and $\mathbf{L}$ being usually non-causal low-pass zero-phase-shift filters. The formula (1) represents a uniformed framework for ILC and repetitive control (RC) [7]. The main obstacle in practical implementation is that there are no analytical methods for choosing the effective filtering that

[*]e-mail: bartlomiej.ufnalski@ee.pw.edu.pl

will stabilize the system in the presence of a usually unknown repetitive disturbance. Therefore, filters are often selected by guessing and checking. To ensure sufficient robustness, a low-pass filtering with a cut-off frequency much below the Nyquist limit has to be implemented, which in turn compromises the effectiveness of the controller by reducing considerably its bandwidth. That is why this scheme has not gained much acceptance among power electronic practitioners whose converters are usually subject to unknown repetitive load currents of high harmonic content, including harmonics non-rejectable due to plant limitations, and are expected to perform even hundreds of millions repetitions without resetting. Nevertheless, the classic ILC algorithm robustified using various $\mathbf{Q}$ and $\mathbf{L}$ design strategies offers high-performance practical position control schemes and as such becomes more and more popular among motion control engineers [8–11] whose mechanical repetitive systems are subject to often negligible exogenous, i.e. unknown, repetitive disturbance forces, which makes the design requirements easier to define.

It should be noted that (1) with $\mathbf{Q} = \mathbf{L} = 1$ constitutes the model of any repetitive signal and as such places itself within the context of the internal model principle (IMP). The same IMP is utilized to synthesize multi-resonant (multi-oscillatory) controllers for selective harmonic disturbance rejection. By contrast, the multi-oscillatory controllers do not suffer from long term stability problems and thus are one of the best alternatives for a repetitive control of high-performance CACF inverters [12–14]. The multi-oscillatory controllers have also their limitations related to the problematic implementation of oscillatory terms near the controller bandwidth and the computational burden growing with the number of harmonics needed to be rejected. They are also sensitive to phase lags and in high-performance converters it is required to take special measures to compensate for these delays [15].

Recently, two new soft-computing approaches to repetitive control have been proposed. In the first one an artificial neural network (ANN) is used to shape the optimal control signal in the iterative manner [16] and should not be confused with the B-spline based voltage controller reported in [17] that employs the idea scrutinized in [18], i.e. does not take advantage of a global update rule. The polynomial and rational basis functions are sometimes incorporated into repetitive systems to produce a control signal which, i.a., is less prone to overlearning thanks to the smoothing effect and/or introduces inverse dynamics to enhance transient performance. Some of the few studied examples from motion control field are [19–21]. The most recent approach to repetitive process control employs PSO for direct optimization of the control signal in the online mode [1]. The proposed swarm has been modified to cope with a dynamic optimization problem (DOP) brought about by the non-stationarity of the disturbance. In [1] a single-swarm solution is reported. Such a solution results in high dimensionality of the fitness function that makes the on-line search for a good suboptimal repetitive control signal a difficult task. In the single-swarm approach the swarm has to be managed in such a way that its explorative ability is kept at a high level. This in turn results in a slow convergence rate. In order to overcome this difficulty and improve responsiveness of the swarm, without deteriorating its exploitative ability, a multi-swarm repetitive controller is proposed here. The name PDPSRC is then modified into plug-in direct multi-swarm repetitive controller (PDMSRC) to reflect the nature of the population used here. The performance of the controller has been verified through numerical simulation and selected results are shown here to illustrate the possibility of reducing the dimensionality of the problem seen by the separate subswarms without a major loss of output voltage waveform quality.

The main contribution of this paper is the second step towards a versatile swarm controller for repetitive processes – the step in which the responsiveness of the stochastic controller is improved by replacing the single-swarm optimizer used in [1] by its multi-swarm counterpart. It should be highlighted that the particle swarm optimization algorithm is not used here in the offline mode to determine parameters in any of the ILC schemes already reported in the literature. The proposed swarm explicitly stores control signal samples and minimizes a user-defined cost function in the online mode. The optimization task at hand is then equivalent to control task itself. No $\mathbf{Q}$-filtering (indispensable in (1)) is used here. The robustness of the control system is shaped by the appropriate cost function selection – here by incorporating a penalty for excessive control signal dynamics characteristic to the overlearning phenomenon.

2. Plug-in direct multi-swarm repetitive controller

Particle swarms are gaining more and more acceptance within the DOP field. A representative set of swarm movement laws effective in the dynamic environments can be found, e.g., in [22]. The previously developed PDPSRC assumes a single swarm travelling through the search space [1]. This implies that a single particle stores all control signal samples per period of a reference signal as depicted in Fig. 1. In the proposed here PDMSRC, the task of online optimization of a shape of the control signal according to a given performance index has been divided between N independent swarms. This means that the α-dimensional DOP has been split into less dimensional DOPs and each of them has been assigned to a different swarm. In general, the task does not have to be split evenly (Fig. 2). In this study only subswarms equal in their dimensionality have been tested. However, later on some hints are given about the possible advantages of swarms covering particular parts of the control signal – not necessarily equal in the number of samples.

The meanings of *subswarm* and *multi-swarm* terms used throughout this paper should not be confused with their most popular interpretations as, e.g., in [23]. Usually both of them are used in the context of a multimodal optimization (MMO). Here the problem is assumed to be unimodal. The swarm has been divided into subswarms to reduce dimensionality for each swarm, i.e. a single particle from a selected swarm does not carry complete candidate solution any more whereas in a typical multimodal problem each particle in each subswarm stores complete candidate solution. Moreover, in a typical MMO information sharing between subswarms is not totally suppressed whereas here subswarms operate in the mutually exclusive subsets of dimensions and there is no information exchange between them at all. The terms *subswarm* and *swarm* can then be used here interchangeably because each subswarm operates as an independent entity.

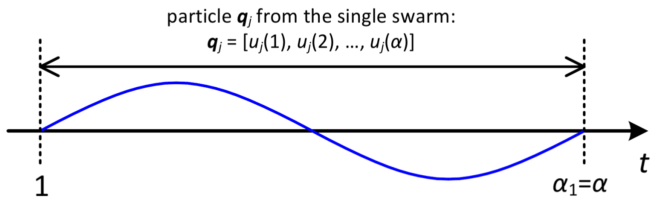

particle q_j from the single swarm:
$q_j = [u_j(1), u_j(2), \ldots, u_j(\alpha)]$
t
1
$\alpha_1 = \alpha$

Fig. 1. The single-swarm approach: a single particle covers all the α samples of the control signal related to the entire period of the depicted reference signal

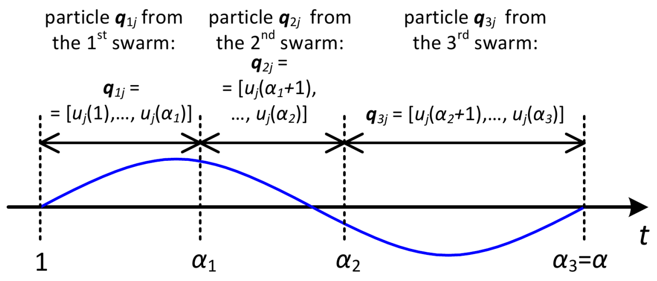

particle q_{1j} from the 1st swarm:
particle q_{2j} from the 2nd swarm:
particle q_{3j} from the 3rd swarm:
$q_{1j} = [u_j(1),\ldots, u_j(\alpha_1)]$
$q_{2j} = [u_j(\alpha_1+1), \ldots, u_j(\alpha_2)]$
$q_{3j} = [u_j(\alpha_2+1),\ldots, u_j(\alpha_3)]$
t
1
α_1
α_2
$\alpha_3=\alpha$

Fig. 2. The multi-swarm approach: a single particle from a subswarm covers only the subset of samples of the control signal related to the entire period of the depicted reference signal – the scenario with 3 swarms and a non-even task distribution

The form of the fitness function (performance index) is identical for all subswarms and is as follows

$$\mathcal{J}(k,n) = \mathcal{J}_0 + \underbrace{\sum_{p=\alpha_{n-1}+1}^{\alpha_n} \left(u_C^{\mathrm{ref}}(p) - u_C^{\mathrm{m}}(p,k)\right)^2}_{\text{penalty for control error}} + \underbrace{\beta \sum_{p=\alpha_{n-1}+2}^{\alpha_n} \left(u_{\mathrm{PSO}}(p,k) - u_{\mathrm{PSO}}(p-1,k)\right)^2}_{\text{penalty for control signal dynamics}}, \tag{2}$$

where k is again the reference signal pass index, p is again the sample index reset at each pass beginning, $n \in [1, N]$ denotes the subswarm identification index, $\alpha_n \in \{\alpha_1, \alpha_2, \ldots, \alpha_N\}$ define junctions between subswarms and β is the subjective penalty factor. It has been assumed that each junction point belongs to the swarm on its left hand side. Also, α_0 is always equal to 0 yielding the beginning of the pass at index $p = 1$, and α_N, where N denotes the number of swarms, is identical with α being the pass length equal to the single period of the reference voltage u_C^{ref}. The positive constant $\mathcal{J}_0$ in (2) is introduced to ensure positive definitiveness of the performance index which is crucial for a knowledge evaporation mechanism described later on in the paper. The superscript in u_C^{m} denotes a measurement signal corrupted by the noise.

Since particles directly store samples of the control signal, they can be represented using vectors as follows

$$\mathbf{q}_{nj} = [\mathbf{u}_j(\alpha_{n-1}+1), \mathbf{u}_j(\alpha_{n-1}+2), \ldots, \mathbf{u}_j(\alpha_n)], \tag{3}$$

where $j \in \{1, 2, \ldots, S\}$, with S being the swarm size, is the particle's identification number within the n-th swarm. In this study all subswarms are equinumerable. The future control signal is constructed from individual solutions by concatenating all NS vectors into a vector

$$\mathbf{u}_i^{\mathrm{PSO}} = [\mathbf{q}_{11}(i), \mathbf{q}_{21}(i), \ldots, \mathbf{q}_{N1}(i), \mathbf{q}_{21}(i), \ldots, \mathbf{q}_{nj}(i), \ldots, \mathbf{q}_{NS}(i)], \tag{4}$$

where i denotes swarm iteration number. The control signal samples generated by the swarm form a time series as follows

$$\mathbf{u}_{\mathrm{PSO}} = \left[\mathbf{u}_1^{\mathrm{PSO}}, \mathbf{u}_2^{\mathrm{PSO}}, \mathbf{u}_3^{\mathrm{PSO}}, \ldots\right], \tag{5}$$

which real-valued entries are equivalent to u_{PSO} in (2) and serve the same purpose as u in (1).

In this study the synchronous update rule is employed, i.e. the subswarms, and thus also $\mathbf{u}_{\mathrm{PSO}}$, are updated after passing all αS consecutive control signal values to the PWM (pulse width modulator). The swarm iteration should not be confused with the reference signal period. It takes S such periods to rate all particles in all subswarms. From the plant's standpoint the update in the k-direction takes place once per αS sampling periods. However, it is possible to distribute in time most PDPSRC/PDMSRC related calculations [24]. For example, (2) can be calculated by adding increments after each sampling time, (6) and (7) require invoking calculations for only one dimension and only one particle in only one subswarm after a given sampling instance. Hence the computational complexity of the algorithm does not grow with an increasing number of subswarms. Synchronization between all time indexes is illustrated in Fig. 3. Their descriptions are collated in Table 1. The resulting controller is depicted in Fig. 4. The PDMSRC is accompanied by the RFF, FSF and RDF that are briefly described in Sec. 3.

Table 1
Integer time indexes

Index	Symbol	Max. value
Pass	k	$+\infty$
Sample	p	α
Subswarm	n	N
Particle	j	S
Swarm iteration	i	$+\infty$

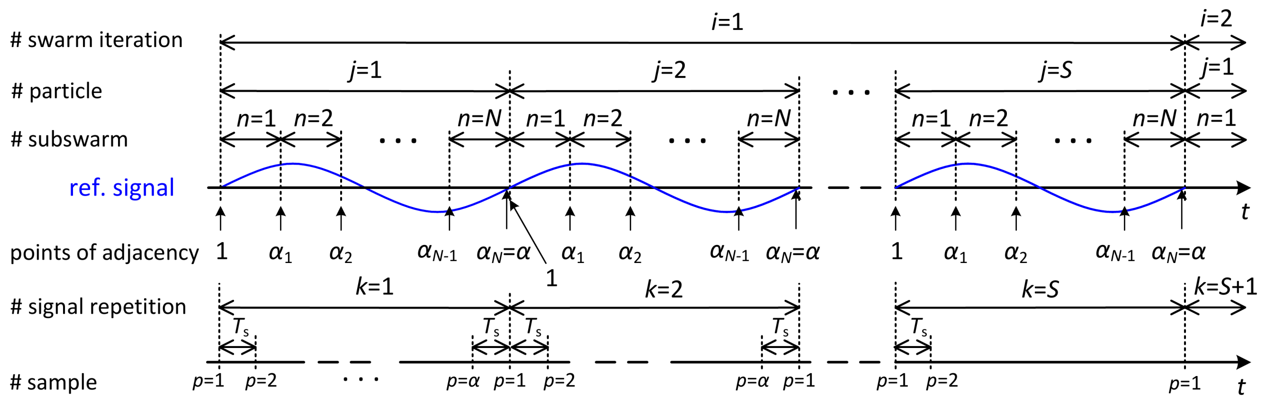
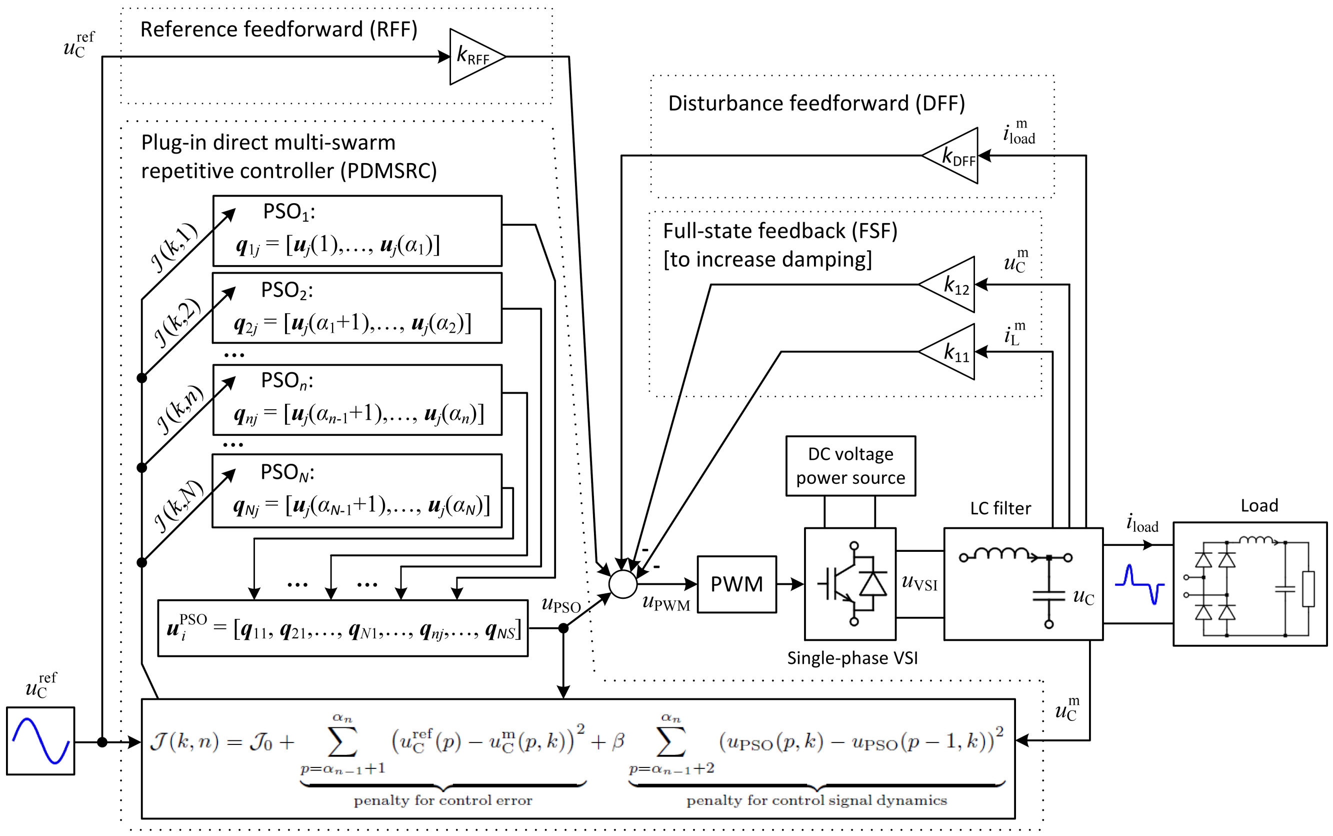


Fig. 3. Synchronization of the time indexes



Reference feedforward (RFF)
kRFF
Disturbance feedforward (DFF)
kDFF
Plug-in direct multi-swarm repetitive controller (PDMSRC)
Full-state feedback (FSF) [to increase damping]
k12
k11
PSO1
PSO2
PSOn
PSON
DC voltage power source
LC filter
Load
PWM
Single-phase VSI
penalty for control error
penalty for control signal dynamics

Fig. 4. A schematic diagram of the control system including a full-state feedback controller and a repetitive disturbance feedforward path – the exemplary nonlinear load depicted for clarity (the block labeled as Load)

The optimization task at hand is of the DOP type due to varying load conditions. Therefore at least two mechanisms have to be implemented – one to keep the swarm alive and another one to gradually forget possibly outdated memories. The simple idea of placing repellers at an already detected *gbest* and stored in the swarm memory *pbest*s [25] has been used. A speed and position update rules are as follows

$$\mathbf{v}_{nj}(i+1) = c_1\mathbf{v}_{nj}(i) + c_2 r^{\text{pbest}}\delta_p\left(\mathbf{q}_{nj}^{\text{pbest}} - \mathbf{q}_{nj}(i)\right) + c_3 r^{\text{gbest}}\delta_p\left(\mathbf{q}_n^{\text{gbest}} - \mathbf{q}_{nj}(i)\right), \tag{6}$$

$$\mathbf{q}_{nj}(i+1) = \mathbf{q}_{nj}(i) + \min\{\max\{-v_{\text{clmp}}, \mathbf{v}_{nj}\,(i+1)\}, v_{\text{clmp}}\}, \tag{7}$$

where $\mathbf{v}_{nj}$ and $\mathbf{q}_{nj}$ are the velocity and position of the j-th particle within the n-th subswarm, $\mathbf{q}_{nj}^{\text{pbest}}$ stores the best solution proposed so far by the j-th particle from the n-th subswarm, $\mathbf{q}_n^{\text{gbest}}$ denotes the best solution found so far by the n-th subswarm, c_1, c_2 and c_3 are the inertia, cognitive and social weights, respectively. A velocity clamping is implemented and the maximum speed is v_{clmp}. The random numbers r^{pbest} and r^{gbest} are uniformly distributed in the unit interval. In all experiments described in this paper, the c_1, c_2 and c_3 factors have been calculated using the constricted PSO formula [26] and are 0.73, 0.73·2.05 and 0.73·2.05, respectively. The direction variable δ_p, having value of -1 or 1, enables the swarms to switch between attract and repel modes and is chosen to be dimension-wise (p-wise), i.e. an individual control of diversity is possible in each search dimension. The Euclidean radius has been selected as the diversity measure

$$D^{p-\text{wise}}_{\text{radius}}(p) = \frac{1}{2}(\max\{q_{n1}(p), \ldots, q_{nS}(p)\} - \min\{q_{n1}(p), \ldots, q_{nS}(p)\}) \tag{8}$$

and the attract and repel modes are being chosen according to a user-defined diversity threshold D_{thold} that represents the trade-off between a steady-state control error and a controller dynamics in the pass-to-pass direction, and has to be chosen by guessing and checking.

The potentially outdated memory of the *pbest*s, and at the same time of the *gbest* being the best of all *pbest*s, is handled using the knowledge gradual evaporation concept [27]. This mechanism forces particles to loose gradually their personal best fitness P_{nj} according to the following rule

$$\begin{bmatrix} P_{nj}(i+1) \\ \mathbf{q}^{\text{pbest}}_{nj} \end{bmatrix} = \begin{cases} \begin{bmatrix} \rho P_{nj}(i) \\ \mathbf{q}^{\text{pbest}}_{nj} \end{bmatrix} & \text{if } \mathcal{J}(\mathbf{q}_{nj}(i+1)) \geq \rho P_{nj}(i) \\ \begin{bmatrix} \mathcal{J}(\mathbf{q}_{nj}(i+1)) \\ \mathbf{q}_{nj}(i+1) \end{bmatrix} & \text{if } \mathcal{J}(\mathbf{q}_{nj}(i+1)) < \rho P_{nj}(i), \end{cases} \tag{9}$$

where ρ has a positive value bigger than 1 for any positive-definite functional $\mathcal{J}$ and an optimization task formulated as the minimization one. The smaller the value of ρ, the slower the transition to the new optimum after a change in the shape of the load current whereas too big a value of ρ is detrimental to the output voltage quality under the repetitive disturbance due to too fast an evaporation of good solutions that in this particular situation do not become outdated. The evaporation constant has to be set by the trial and error method. It should be noticed that the knowledge evaporation mechanism (9) does not work for a positive semi-definite problem, because a zero-valued solution cannot be "forgotten" by multiplying it by ρ. Though it is highly unlikely that the sum of squared errors and increments present in (2) would reduce to zero in any physical system, it is still possible to get zero value from a theoretical point of view. A positive offset $\mathcal{J}_0$ in (2) is then added for mathematical elegance. Its value along with ρ shapes the forgetting process. Here a relatively small value of 0.01 has been assumed which is practically negligible and the forgetting is almost identical as for the sole sum of squares. Key parameters of the swarm are collated in Table 2. A flowchart of the swarm repetitive control algorithm is depicted in Fig. 5.

Table 2
Parameters of the swarm

Parameter	Symbol	Value
Dimensionality of the problem	α	200
Number of particles[a]	S	25
Swarms' update frequency	$f^{\text{ref}} \cdot S^{-1}$	2 Hz
Number of subswarms[b]	N	1, 2, 5, 10, 20 or 50
Points of division[c] (for $\alpha_0 = 0$)	α_n	$n\frac{\alpha}{N} \wedge n \in \{1, 2, \ldots, N\}$
Evaporation constant[d]	ρ	1.05, 1.10, 1.20, 1.40 or 1.50
Diversity threshold	D_{thold}	$1.5 \cdot 325^{-1}$
Penalty factor	β	0.25
Constant summand in cost function	$\mathcal{J}_0$	0.01
Velocity clamping level	v_{clmp}	9.0

[a] identical for all subswarms
[b] selected case scenarios with subswarms that cover $\frac{\alpha}{N}$ dimensions each
[c] in this study evenly spaced throughout the period of the reference signal
[d] exemplary values – always specified in captions

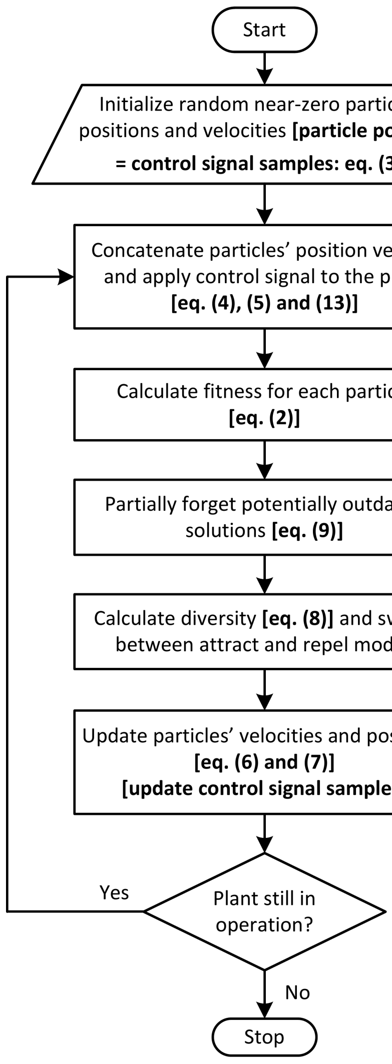

Start
Initialize random near-zero particles' positions and velocities [particle position = control signal samples: eq. (3)]
Concatenate particles' position vectors and apply control signal to the plant [eq. (4), (5) and (13)]
Calculate fitness for each particle [eq. (2)]
Partially forget potentially outdated solutions [eq. (9)]
Calculate diversity [eq. (8)] and switch between attract and repel modes
Update particles' velocities and positions [eq. (6) and (7)] [update control signal samples]
Plant still in operation?
Yes
No
Stop

Fig. 5. A flowchart of the swarm repetitive controller

The two above-mentioned DOP-enabling mechanisms are pivotal to correct operation of the controller and so the evaporation constant and the diversity threshold are two parameters critical for the correct operation of the PSO. Also the velocity clamping level is useful in shortening the settling time in the k-direction. At the moment, all of them have to be tuned by guessing and checking. Fortunately, they have very straightforward interpretation and the optimization/control results are not very sensitive with respect to changes in these settings, i.e. the change of 10% in the evaporation constant, or the change of 50% in the diversity threshold or the velocity clamping level do not render the algorithm ineffective. These settings constitute a compromise between the responsiveness (the ability to react fast enough to environment dynamism caused by load variations) and the steady-state voltage quality. The bigger the evaporation constant and the diversity threshold, the faster the transition to a new optimum. However, the bigger they are, the weaker is the swarm's ability to exploit the search space and hence more distorted is the output voltage. The lower the velocity clamping level, the more smooth the transition to a new optimum. Nevertheless, this happens at the cost of the transition duration, i.e. the responsiveness.

3. Full-state feedback controller and disturbance feedforward

Clearly, the plug-in repetitive controller acts only in the k-direction. Such a controller has to be assisted by a non-repetitive controller acting in the p-direction. For the purpose of this study, the full-state feedback (FSF) has been implemented to increase damping in the highly underdamped plant (compare R_f with R_crit in Table 3). This gives control signal

$$u_\mathrm{FSF} = -(k_{11} i_\mathrm{L}^\mathrm{m} + k_{12} u_\mathrm{C}^\mathrm{m}), \tag{10}$$

additive to the PDMSRC output signal. In this particular study the damping has been increased 5 times, i.e. FSF gains k_{11} and k_{12} have been determined using the pole placement procedure to shift closed-loop poles 5 times deeper into the left-half s-plane in respect to open-loop poles. The standard reference feedforward (RFF) path

$$u_\mathrm{RFF} = (1 + k_{12}) u_\mathrm{C}^\mathrm{ref}, \tag{11}$$

gives a unity gain for the zero frequency [28]. Also, the disturbance static feedforward (DFF) path is introduced to compensate the resistive voltage drop (for the zero frequency) [13]

$$u_\mathrm{DFF} = (\widehat{R}_\mathrm{f} + k_{11}) i_\mathrm{load}^\mathrm{m}, \tag{12}$$

where $\widehat{R}_\mathrm{f}$ is the identified resistance of the output filter and $i_\mathrm{load}^\mathrm{m}$ denotes the measured load current. A relatively high identification error is assumed in this study ($\widehat{R}_\mathrm{f} = 0.5 R_\mathrm{f}$) to emphasize influence of the repetitive controller. Also the prediction of the load current based on the previous pass proposed in [29] to compensate the overall lag caused by a digital implementation of the controller and an inherent delay of the PWM has been omitted here in order to produce a more significant control error for the PDMSRC and as a result to make the case scenario more illustrative. The resulting control signal passed to the modulator

$$u_\mathrm{PWM} = u_\mathrm{PSO} + u_\mathrm{RFF} + u_\mathrm{FSF} + u_\mathrm{DFF} \tag{13}$$

acts simultaneously in the along the pass direction and the pass to pass direction.

Table 3
Parameters of the converter

Parameter	Symbol	Value
Filter inductance	L_f	300 μH
Filter capacitance	C_f	160 μF
Filter resistance	R_f	0.2 Ω
Filter resonant frequency	f_res	726 Hz
Critical damping resistance	R_crit	2.74 Ω
Reference frequency	f^ref	50 Hz
Sampling/PWM frequency	f_s	10 kHz
Pass length	α	200
DC-link voltage	k_c	450 V
Measurement noise	–	ca. 1%
Voltage measurement gain	k_u	1/325 [1/V]
Current measurement gain	k_i	1/200 [1/A]
Rectifier power	–	ca. 6 kW
Rectifier current crest factor	–	ca. 2.5
Resistive load power	–	ca. 4 kW

4. Numerical experiment results

Selected parameters of the plant are given in Table 3. A test scenario is as follows:

a) the swarms are initialized with near zero $\mathbf{u}_0^\mathrm{PSO}$ control vector (no pre-tuning, e.g. for no load conditions, is assumed),
b) the resistive load of ca. 4 kW is applied for 200 s, 100 s or 50 s adequately to the swarm dynamics being evaluated,
c) the resistive load is switched off and the diode rectifier (ca. 6 kW, current crest factor of ca. 2.5) is switched on for 200 s, 100 s or 50 s adequately to the swarm dynamics being evaluated,
d) the diode rectifier is switched off and the initial resistive load is applied once again.

It is apparent from Fig. 6 that the 2-swarm approach is more effective than the single-swarm in terms of convergence rate without reducing the capability to explore effectively the search space. All swarm movement parameters, such as D_thold, v_clmp and ρ have been left unchanged. The performance improvement comes from problem dimensionality reduction. It should be noted that the multi-swarm algorithm does not imply higher computational burden than the single-swarm one. The amount of data to be stored and processed is almost identical. It has been tested whether further increasing of the number of subswarms is clearly beneficial. Certainly, at some point further division of the optimization task can be even detrimental. For example, in the boundary case of single-dimensional swarms, i.e. for the 1D optimization landscape, it is impossible to calculate second term in (2) and this term is required to prevent overlearning that could lead to instability in the long horizon – an issue similar to the phenomenon encountered in the classic ILC scheme. The performance of the 2-swarm and 5-swarm controllers has been compared in Fig. 7. Potential benefits of the 5-swarm algorithm over 2-swarm one are disputable if identical relatively low evaporation rates ρ are assumed. However, the gradual fitness evaporation affects the quality of the output voltage uniquely for different optimization landscapes. It has been observed that faster knowledge evaporation is less deleterious if more subswarms are used. This is illustrated in Fig. 8. In this particular setup the 2-swarm controller is unable to track effectively the moving optimum in the noisy environment whereas its 5-swarm counterpart does the job effectively. The root mean square error (RMSE) as a cumulative error indicator does not provide comprehensive information on voltage quality. Therefore, the output voltage is always scrutinized using instantaneous error graphs. An example of such a graph for the 5-swarm controller is given in Fig. 9. Also, the transient states caused by load type change have been monitored visually in graphs depicting the evolution of output voltage in the k-direction as in Fig. 10.

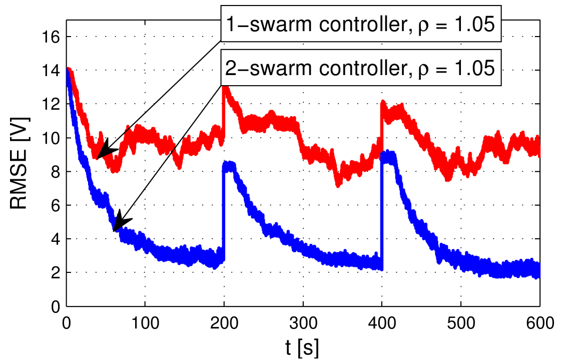


Fig. 6. Comparison of the RMSE graphs for the single-swarm and 2-swarm controllers

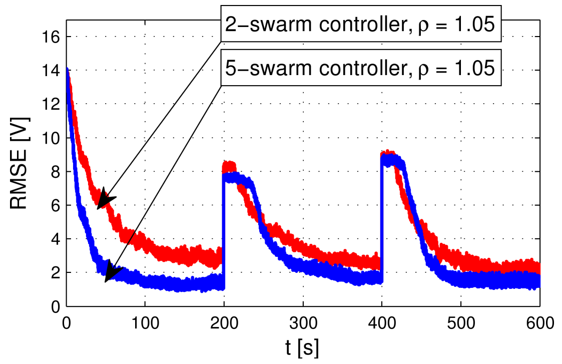


Fig. 7. Comparison of the RMSE graphs for the 2-swarm and 5-swarm controllers

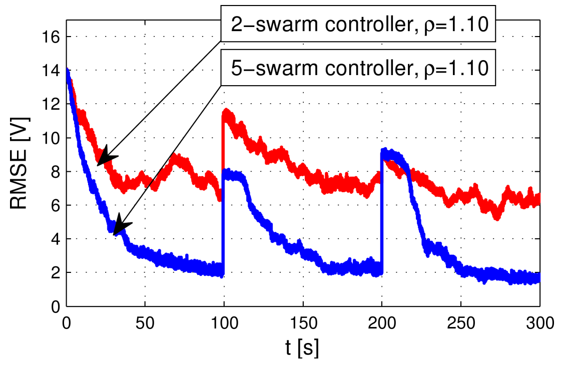


Fig. 8. Comparison of the RMSE graphs for the 2-swarm and 5-swarm controllers in the case of fast knowledge evaporation ($\rho = 1.10$)

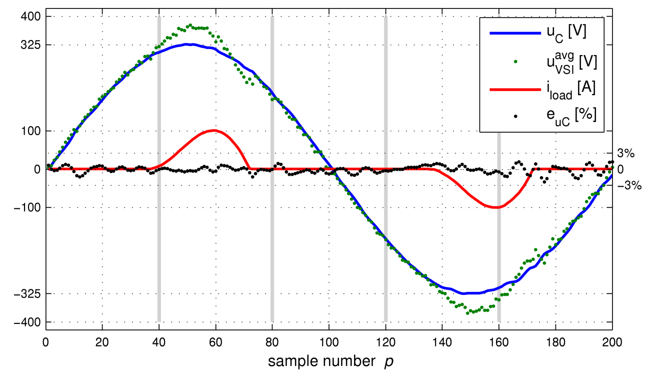


Fig. 9. The shape of the output voltage under the diode rectifier load for the 5-swarm controller – the load current, the commanded average voltage for the VSI and the control error added for clarity

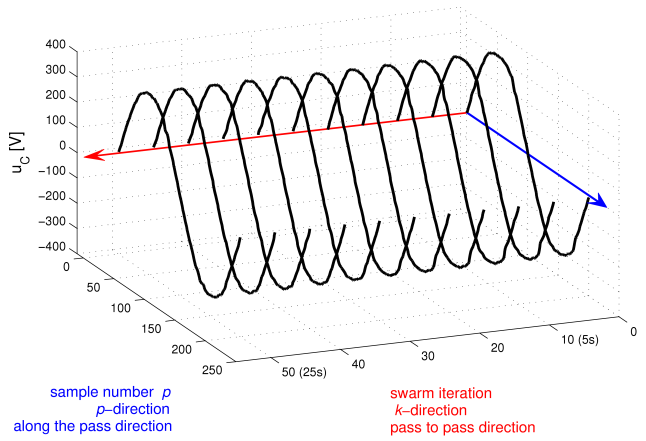


Fig. 10. The evolution of the output voltage after connecting the nonlinear load (the diode rectifier) for the 5-swarm repetitive controller with $\rho = 1.10$

Only one minor disadvantage of the multi-swarm approach in comparison with the single-swarm one has been identified during the study. It sometimes happens that the control signal at transition points between subswarms has clearly higher increment in the p-direction than the signal proposed by a single swarm. This is due to the lack of direct communication between the swarms. The only interaction between optimizers is through the physical plant itself. This implies that a given swarm strives to maximize its performance at the cost of neighbours. However, this seems to be manageable taking into account an overall quality of the output voltage waveform, i.e. an acceptable tradeoff between the number of transition points and the convergence rate can be worked out by the trial and error method. This has been illustrated in Figs. 11–15 and 16, respectively. The less dimensional search subspace is, i.e. the more swarms operate in parallel, the faster knowledge evaporation can be applied due to a simpler landscape. This, in turn, allows for a faster responsiveness of the swarm when a shape of the load current changes. The PDMSRC can effectively search for the optimal control signal even in the near-extreme case of 50 swarms; however, the transitions between swarms are not always quite smooth. This case has been illustrated in Fig. 16 with vertical gray bars indicating subsets of dimensions searched through by the subswarms (to be compared with e.g. Fig. 12).

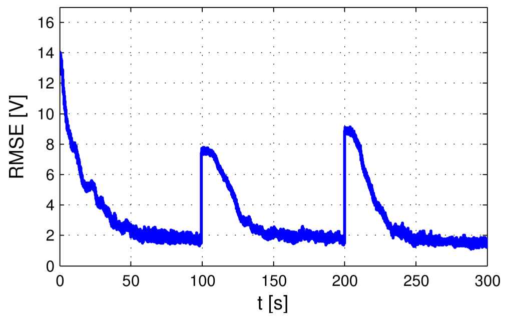


Fig. 11. Evolution of the RMSE for the optimization space divided into 20D subspaces, i.e. 10-swarm controller, and $\rho = 1.20$

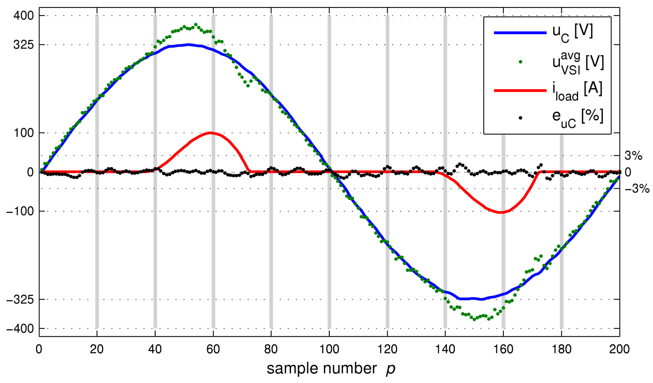


Fig. 12. The shape of the output voltage under the diode rectifier load for the 10-swarm controller and $\rho = 1.20$ – the load current, the commanded average voltage for the VSI and the control error added for clarity

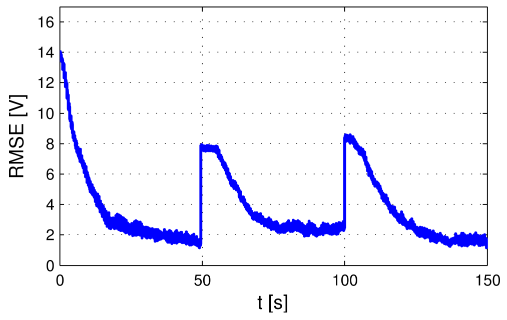


Fig. 13. Evolution of the RMSE for the optimization space divided into 10D subspaces, i.e. 20-swarm controller, and $\rho = 1.40$

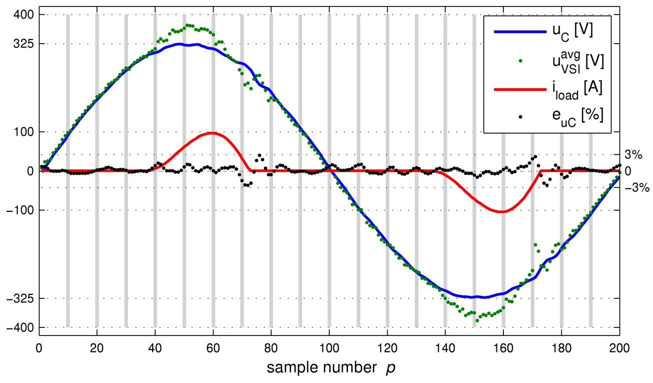


Fig. 14. The shape of the output voltage under the diode rectifier load for the 20-swarm controller and $\rho = 1.40$ – the load current, the commanded average voltage for the VSI and the control error added for clarity

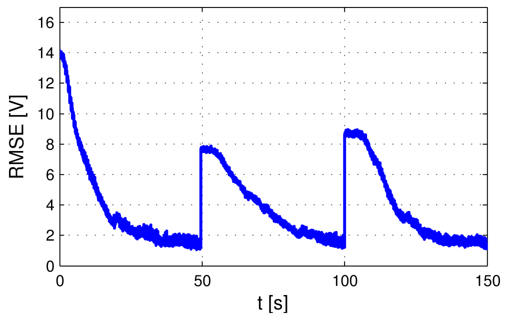


Fig. 15. Evolution of the RMSE for the "extreme" case of optimization space divided into 4D subspaces, i.e. 50-swarm controller, and $\rho = 1.50$

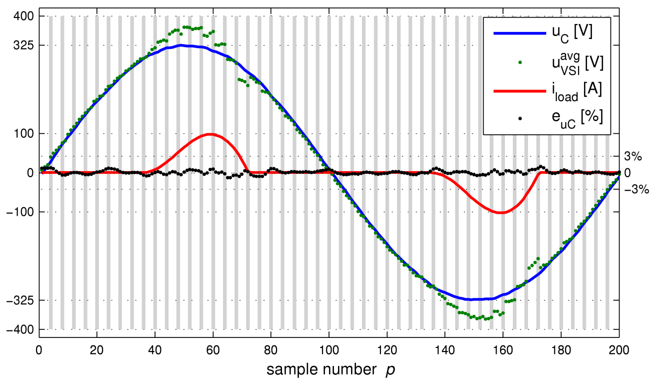


Fig. 16. The shape of the output voltage under the diode rectifier load for the 50-swarm controller and $\rho = 1.50$ – the load current, the commanded average voltage for the VSI and the control error added for clarity

The multi-swarm approach is clearly beneficial with respect to the convergence rate which is apparent from Figs. 8, 11, 13 and 15. These results also indicate that under the steady nonlinear load conditions the output voltage quality is very similar in terms of the RMSE performance index in all four cases. The accompanying Figs. 9, 12, 14 and 16 also demonstrate similar quality of the voltage waveforms at a steady state. It can be read from the graphs that for this particular plant going beyond 20 subswarms does not practically improve the convergence rate any further. The recommendation is then to synthesize such a controller using 10–20 subswarms.

It should be recalled that the subswarms do not necessarily have to cover identical number of dimensions. It would be beneficial to avoid transitions between swarms at samples with high absolutes values of the load current derivative. However, this has been assumed to be out of the scope of the paper and probably such an approach would be of no practical use due to the lack of a priori knowledge about the shape of a load current needed to determine optimal adjacent points for the swarms.

5. Responsiveness in the pass-to-pass direction

As illustrated in Sec. 4, the proposed swarm controller is relatively slow in the k-direction. However, it is important to acknowledge that the controller action in the k-direction is not designed to stabilize the system and shape the transient states on a sample-after-sample basis. This has to be rendered by the non-repetitive part of the controller (here RFF, FSF and DFF). The PDMSRC performs only fine shaping of the control signal from pass to pass. Obviously, its dynamics has to match the anticipated frequency of disturbance load variations. For the proposed controller, if significant load current shape variations occur for periods of time shorter than ca. 20 s, the benefits are disputable. The reaction time in the k-direction is bounded to be at the level of tens of seconds if the PSO approach is utilized. It is widely recognized that usually, i.e. for most of practical optimization problems, a sufficient number of particles is around 30 and that an effective compromise between exploration and exploitation requires hundreds or even thousands of swarm iterations. It has been tested that small swarms (of around 5 particles) do not perform well in the proposed control system. The 25-particle swarm has been identified as the effective one and then selected as the base for this study. This in turn implies that for the 50 Hz reference signal the swarms are iterated at 2 Hz. Since, even for fitness landscapes with a reduced dimensionality, several tens of iterations are needed to move the swarm effectively near a new optimum, response times counted in tens of seconds seem to be inevitable for the synchronous update rule used here. Nevertheless, to the best of authors' knowledge there are systems that are characterized by load profiles changing far slower than on the tens of seconds scale. Additionally, presented numerical experiments demonstrate that the multi-swarm approach is more effective than the single-swarm one with respect to its response time. This concept could be potentially helpful also in other on-line optimization, e.g. estimation by optimization, problems such as self-commissioning of electric drives, which are time/iteration-critical.

It should be highlighted that in general the proposed multi-swarm optimizer is not equivalent to its single-swarm counterpart with respect to the optimal solution to be found. This is due to the coupled nature of most real-life optimization tasks which in turn implies that a problem with α decision variables

$$\mathcal{J} = f(u_1, u_2, \ldots, u_\alpha) \tag{14}$$

cannot be easily (or at all) split into an equivalent set of N subproblems

$$\begin{cases} \mathcal{J}_1 = f_1(u_1, \ldots, u_{\alpha_1}) \\ \mathcal{J}_2 = f_2(u_{\alpha_1+1}, \ldots, u_{\alpha_2}) \\ \quad\vdots \\ \mathcal{J}_N = f_N(u_{\alpha_{N-1}+1}, \ldots, u_\alpha). \end{cases} \tag{15}$$

However, most real-life optimization problems in control systems are based on user-defined performance indices. It is the engineer who designs fitness functions that are appropriate to the problem at hand, i.e. that force a desired behaviour of the system. This means that the problem is not necessarily bound to be defined as (14). It has been illustrated with CACF VSI that for the online swarm-based optimization it is beneficial to redefine the problem into the form of (15) and its lack of equivalence to (14) occurs to be of little importance.

6. Practicalities

Optimal control is usually associated with optimal offline tuning of controller gains. Computational burden of an optimization algorithm is then of next to no importance as long as the procedure can be completed in a reasonable time. By contrast, in any online optimization task the computational complexity of the algorithm becomes the major area of concern. In the proposed PDMSRC, the PSO itself constitutes the control algorithm and hence all the PSO related calculations have to be performed in real time. In large part the choice of this particular optimization algorithm, as well as all necessary modifications required to handle the dynamic nature of the discussed optimization task, has been dictated by their practicality in terms of real-time implementation on an off-the-shelf microcontroller. The execution of the PDMSRC code can be distributed along all αS sample periods as illustrated in [30]. Moreover, the complexity of the algorithm does not grow with the number of subswarms because the more subswarms are introduced, the lower the dimensionality of the particle becomes as shown in [31]. It has been tested that the computational power of, e.g., the industrial microcontroller TMS320F2812 is sufficient to execute the swarm algorithm featuring parameters given in Table 2 [24]. Regardless of the number of subswarms, the amount of time necessary to complete all calculations on this particular digital signal controller is less than 30 μs. The physical implementation of the controller is the subject of our current work.

7. Conclusions

A multi-swarm approach to a direct particle swarm repetitive controller has been proposed and investigated. It has been shown that the previously developed plug-in direct single-swarm repetitive controller for the single-phase inverter with the LC output filter can be enhanced by applying the multi-swarm concept. The optimization task has been divided into several less dimensional subtasks, which occurs to simplify the search. By doing so, the convergence rate of the swarm is improved without a significant loss of output voltage quality. The obtained numerical results suggest feasibility of the developed algorithm for controlling the continuous repetitive process of PWM VSI output voltage shaping. The proposed algorithm is universal in the sense that it can serve as a plug-in repetitive controller for any continuous repetitive process, as well as for batch processes with a state resetting capability at the beginning of each trial.

Acknowledgements. The research was partially supported by the statutory fund of Electrical Drive Division within the Institute of Control and Industrial Electronics, Faculty of Electrical Engineering, Warsaw University of Technology, Poland.

REFERENCES

[1] B. Ufnalski and L.M. Grzesiak, "A plug-in direct particle swarm repetitive controller for a single-phase inverter", *Electrical Review (Przegląd Elektrotechniczny)* 90 (6), 6–11 (2014).

[2] R.W. Longman, "Iterative/repetitive learning control: learning from theory, simulations, and experiments", in *Encyclopedia of the Sciences of Learning*, pp. 1652–1657, Springer, New York, 2012.

[3] Z. Cai, "Iterative learning control: algorithm development and experimental benchmarking", *Ph.D. Thesis*, University of Southampton, Southampton, 2009.

[4] Y. Shi, "Robustyfication in repetitive and iterative learning control", *Ph.D. Thesis*, Columbia University, Columbia, 2013.

[5] E. Rogers, K. Galkowski, and D.H. Owens, "Two decades of research on linear repetitive processes part I: theory", *Int. Workshop on Multidimensional Systems (nDS)* 1, 1–6 (2013).

[6] E. Rogers, K. Galkowski, W. Paszke, and D.H. Owens, "Two decades of research on linear repetitive processes part II: applications", *Int. Workshop on Multidimensional Systems (nDS)* 1, 1–6 (2013).

[7] Y. Wang, F. Gao, and F.J. Doyle III, "Survey on iterative learning control, repetitive control, and run-to-run control", *J. Process Control* 19 (10), 1589–1600 (2009).

[8] Z. Cai, C. Freeman, E. Rogers, and P. Lewin, "Reference shift iterative learning control for a non-minimum phase plant", *American Control Conf. (ACC)* 1, 558–563 (2007).

[9] M. Heertjes, R. Rampadarath, and R. Waiboer, "Nonlinear Q-filter in the learning of nano-positioning motion systems", *Eur. Control Conf. (ECC)* 1, CD-ROM (2009).

[10] I. Rotariu, M. Steinbuch, and R. Ellenbroek, "Adaptive iterative learning control for high precision motion systems", *IEEE Trans. on Control Systems Technology* 16 (5), 1075–1082 (2008).

[11] M. Heertjes and T. Tso, "Nonlinear iterative learning control with applications to lithographic machinery", *Control Engineering Practice* 15 (12), 1545–1555 (2007).

[12] G. Escobar, P. Mattavelli, M. Hernandez-Gomez, and P.R. Martinez-Rodriguez, "Filters with linear-phase properties for repetitive feedback", *IEEE Trans. on Industrial Electronics* 61 (1), 405–413 (2014).

[13] A. Kaszewski, B. Ufnalski, and L.M. Grzesiak, "An LQ controller with disturbance feedforward for the 3-phase 4-leg true sine wave inverter", *IEEE Int. Conf. on Industrial Technology (ICIT)* 1, 1924–1930 (2013).

[14] B. Ufnalski, A. Kaszewski, and L.M. Grzesiak, "Particle swarm optimization of the multioscillatory LQR for a three-phase four-wire voltage-source inverter with an LC output filter", *IEEE Transactions on Industrial Electronics* 62 (1), 484–493 (2015).

[15] G. Escobar, M. Hernandez-Gomez, G.A. Catzin, P.R. Martinez-Rodriguez, and A.A. Valdez-Fernandez, "Implementation of repetitive controllers subject to fractional delays", *Annual Conf. IEEE Industrial Electronics Society (IECON)* 1, 5983–5988 (2013).

[16] B. Ufnalski and L.M. Grzesiak, "Particle swarm optimization of an online trained repetitive neurocontroller for the sine-wave inverter", *Annual Conf. IEEE Industrial Electronics Society (IECON)* 1, 6001–6007 (2013).

[17] H. Deng, R. Oruganti, and D. Srinivasan, "Neural controller for UPS inverters based on B-spline network", *IEEE Trans. on Industrial Electronics* 55 (2), 899–909 (2008).

[18] Y.Q. Chen, K. Moore, and V. Bahl, "Learning feedforward control using a dilated B-spline network: frequency domain analysis and design", *IEEE Trans. on Neural Networks* 15 (2), 355–366 (2004).

[19] J. van de Wijdeven and O. Bosgra, "Stabilizability, performance, and the choice of actuation and observation time windows in iterative learning control", *IEEE Conf. on Decision and Control* 1, 5042–5047 (2006).

[20] J. van de Wijdeven and O.H. Bosgra, "Using basis functions in iterative learning control: analysis and design theory", *Int. J. Control* 83 (4), 661–675 (2010).

[21] J. Bolder and T. Oomen, "Rational basis functions in iterative learning control—with experimental verification on a motion system", *IEEE Trans. on Control Systems Technology* 23 (2), 722–729 (2015).

[22] T. Blackwell, J. Branke, and X. Li, "Particle swarms for dynamic optimization problems", *Swarm Intelligence, Natural Computing Series*, pp. 193–217, Springer, Berlin, 2008.

[23] A.P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, Wiley, London, 2005.

[24] P. Biernat, B. Ufnalski, and L.M. Grzesiak, "Direct particle swarm repetitive controller with time-distributed calculations for real time implementation", *IEEE Int. Conf. on Intelligent Systems (IS)* 1, DOI: 10.1007/978-3-319-11313-5_44 (2014).

[25] J. Riget and J.S. Vesterstrøm, "A diversity-guided particle swarm optimizer – the ARPSO", *Technical Report*, Aarhus Universitet, Denmark, 2002.

[26] R.C. Eberhart, Y. Shi, and J. Kennedy, *Swarm Intelligence*, Morgan Kaufmann Publishers, Berlin, 2001.

[27] X. Cui, J.S. Charles, and T.E. Potok, "A simple distributed particle swarm optimization for dynamic and noisy environments", *Int. Workshop on Nature Inspired Cooperative Strategies for Optimization (NICSO)* 1, 89–102 (2009).

[28] G. Franklin, D. Powell, and M. Workman, *Digital Control of Dynamic Systems*, Prentice Hall, New Jersey, 1997.

[29] B. Ufnalski and L.M. Grzesiak, "Feedback and feedforward repetitive control of single-phase UPS inverters – an online particle swarm optimization approach", *Technical Report*, Scientific Reports of the Cologne University of Applied Sciences, Cologne, 2014.

[30] B. Ufnalski and P. Biernat, "Time-distributed particle swarm repetitive control algorithm", *MATLAB Central*, http://www.mathworks.com/matlabcentral/fileexchange/48967-time-distributed-particle-swarm-repetitive-control-algorithm (2014).

[31] B. Ufnalski, "Plug-in direct particle swarm repetitive controller", *MATLAB Central*, http://www.mathworks.com/matlabcentral/fileexchange/47847-plug-in-direct-particle-swarm-repetitive-controller (2014).