ARCHIVES OF ELECTRICAL ENGINEERING VOL. 63(4), pp. 635-646 (2014)

DOI 10.2478/aee-2014-0044

A performance study on synchronous and asynchronous update rules for a plug-in direct particle swarm repetitive controller

BARTLOMIEJ UFNALSKI, LECH M. GRZESIAK

Faculty of Electrical Engineering, Warsaw University of Technology
Institute of Control and Industrial Electronics
Koszykowa 75, Warszawa 00-662, Poland
tel.: +48 22 234 6138, fax: +48 22 234 6023
e-mail: bartlomiej.ufnalski@ee.pw.edu.pl
URL: http://www.ee.pw.edu.pl

(Received: 15.07.2014, revised: 22.09.2014)

Abstract: In this paper two different update schemes for the recently developed plug-in direct particle swarm repetitive controller (PDPSRC) are investigated and compared. The proposed approach employs the particle swarm optimizer (PSO) to solve in on-line mode a dynamic optimization problem (DOP) related to the control task in the constant-amplitude constant-frequency voltage-source inverter (CACF VSI) with an LC output filter. The effectiveness of synchronous and asynchronous update rules, both commonly used in static optimization problems (SOPs), is assessed and compared in the case of PDPSRC. The performance of the controller, when synthesized using each of the update schemes, is studied numerically.

Key words: repetitive process control, particle swarm optimization, synchronous and asynchronous update rules, dynamic optimization problem, repetitive disturbance rejection, optimal control

Nomenclature

PSO – Particle Swarm Optimization; SUR – Synchronous Update Rule; ASUR –Asynchronous Update Rule; RC – Repetitive Control (-ler); 2D – two-dimensional (system); ILC – Iterative Learning Control (-ler); *k*-direction – pass-to-pass direction; *p*-direction –along the pass direction; PDPSRC – Plug-in Direct Particle Swarm Repetitive Controller (the proposed concept); *p*-wise – calculated for a given search dimension; IMP – Internal Model Principle; DOP – Dynamic Optimization Problem; SOP – Static Optimization Problem; RMSE – Root Mean Square(d) Error; AUC – the Area Under the Curve; VSI – Voltage-Source Inverter; CACF – Constant-Amplitude Constant-Frequency (sine wave); FSF – Full-State Feedback (controller); RFF – Reference Feed-Forward path; DFF – Disturbance Feed-Forward path.

1. Introduction

In many industrial systems the repetitiveness of a control signal is clearly apparent. A family of power electronic converters used to produce CACF sinusoidal voltage may serve as an example. The goal is then to maintain a high quality output voltage despite nonlinear load, i.e. in the presence of non-sinusoidal disturbance currents. This task is known to be tackled using one of the following approaches:

a) by neglecting the repetitiveness of the process at hand and accepting moderate quality of the output voltage under nonlinear loads,
b) by introducing an internal model of anticipated dominant disturbance frequencies – solution known as a multi-resonant or multi-oscillatory controller [1],
c) by introducing a universal internal model of any periodic signal – solution known as ILC [2],
d) by solving in on-line mode, i.e. in real time, the DOP aimed at shaping an optimal control signal.

None of the above techniques provides an ultimate solution. The approach a) often compromises output voltage quality for relatively low computational complexity and non-demanding memory requirements. The solution b) provides selective rejection of disturbance harmonics and is probably the most popular RC scheme in power electronic converters if a high quality voltage is demanded. The main difficulty with that scheme comes from problematic implementation of oscillatory terms near the Nyquist limit, therefore the resulting controller has usually one order of magnitude lower bandwidth than the sampling employed. Another approach is to use ILC technique, which also belongs to the IMP family, to handle internal generation of any desired periodic signal. In ILC this signal is shaped in the iterative manner from pass to pass, i.e. information on control errors from the previous pass is used to correct control signal in the current pass. The system is then analyzed as 2D one and the control signal consists of two addends – one calculated in the p-direction and one calculated in the k-direction. However, the main obstacle in practical implementation is that the ILC has been identified to suffer from long term stability problems [3, 4]. Many research teams put significant effort into enhancing this technique to make it robust. The most common modification to the classic ILC is trying to stop learning for higher frequencies. Even the very basic P-type control law, where u denotes the control signal, e stands for the control error and k_{RC} represents the controller gain, is bound to be modified into

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

by applying low-pass zero-phase-shift filters $\mathbf{Q}$ and $\mathbf{L}$ to stabilize the system. The idea behind (1) is to cut off totally the learning for "prohibited" frequencies. It should be noted that this control law introduces integration of control error in the k-direction. In most practical VSI systems the control error cannot be forced to zero for all shapes of the disturbance current. By the "prohibited" harmonic it is meant any disturbance harmonic that cannot be completely rejected due to a limited DC-link voltage in any VSI. Such a harmonic will have destabilizing impact in the long run. This problem has been already widely acknowledged by practitioners and the finite attenuation in the stopband of any practical digital filter makes it hard to solve (if at all) in infinite time horizon using just the filtering as in (1). That is why this scheme has gained some acceptance mainly among motion control practitioners, e.g. a robotic arm control for an assembly line automation, but has not gained much (if any) such acceptance among power electronics practitioners. This happens due to usually many-fold higher time horizon expressed in number of required stable repetitions in power converters before the system could be reset to avoid significant buildup of high frequencies in the control signal. The emergence of the family d) should be credited to computational power of nowadays digital signal controllers (DSCs). Resources offered by off-the-shelf microcontrollers encourage automation developers to formulate control tasks as DOP ones. Model predictive control (MPC) can serve here as an example. Also repetitive control tasks can be formulated as DOP ones. Feasibility of such solutions has already been demonstrated in the case of neurocontrollers [5-7]. Probably the most recent members of this family use population-based stochastic search methods to shape the control signal optimally by solving in real time in the iterative manner a related DOP. The proposed PDPSRC is such a solution [8, 9].

2. Plug-in direct particle swarm repetitive controller

The most distinguishable feature of the PDPSRC comes from the fact that control signal samples are stored directly in particles. The PSO does not then adapt gains of some other controller but serves, itself, as the controller. This idea is illustrated here using the CACF VSI with an output LC filter (Fig. 1). The objective is to minimize in real time

$$J(k,n) = J_0 + \underbrace{\sum_{p=\alpha_{n-1}+1}^{\alpha_n} (u_C^{\mathrm{ref}}(p) - u_C^{\mathrm{m}}(p,k))^2}_{\text{penalty for control error}} + \beta \underbrace{\sum_{p=\alpha_{n-1}+2}^{\alpha_n} (u_{\mathrm{PSO}}(p,k) - u_{\mathrm{PSO}}(p-1,k))^2}_{\text{penalty for control signal dynamics}}, \tag{2}$$

where n denotes the swarm identification index (the DOP can be divided among several swarms to reduce dimensionality of the landscape seen by each subswarm), β is the penalty factor (determined by guessing and checking), u_C^{ref} is the reference voltage, u_C^{m} denotes the measured voltage, and u_{PSO} represents the control signal in the k-direction plugged into the system already equipped with the p-direction controller. The constant summand J_0 is needed to enable knowledge evaporation also for zero sum of squares. It should be noted that this approach is of model free type and the plant itself is used as the critic. The minimization of J is done by particles which travel through the search space at velocities

$$\mathbf{v}_{nj}(i+1) = c_1\mathbf{v}_{nj}(i) + c_2 r^{\mathrm{pbest}} \delta_p(\mathbf{q}_{nj}^{\mathrm{pbest}}(i) - \mathbf{q}_{nj}(i)) + c_3 r^{\mathrm{gbest}} \delta_p(\mathbf{q}_{n}^{\mathrm{gbest}}(i) - \mathbf{q}_{nj}(i)) \tag{3}$$

and have positions updated according to

$$\mathbf{q}_{nj}(i+1) = \mathbf{q}_{nj}(i) + \min\{\max\{-v_{\mathrm{clmp}}, \mathbf{v}_{nj}(i+1)\}, v_{\mathrm{clmp}}\}, \tag{4}$$

where $\mathbf{v}_{nj}$ and $\mathbf{q}_{nj}$ denotes velocity and position of the j-th particle within the n-th subswarm, $\mathbf{q}_{nj}^{\mathrm{pbest}}$ stores the best solution proposed so far by the j-th particle from the n-th subswarm and

$\mathbf{q}_n^{\text{gbest}}$ is the best solution found so far by the n-th subswarm. The inertia, cognitive and social factors, as well as random numbers present in (3) are set according to standard recipes [8]. The optimization problem is of dynamic type due to varying load conditions. This in turn implies that the population of particles has to be armed with two mechanisms – one that maintains a non-zero diversity and another that deals with outdated memory. The first technique used here introduces repulsion from pbests and gbest (already included in (3)). The direction δ_p is switched from 1 to -1 if the p-wise radius of the swarm deteriorates below a given diversity threshold D_{thold} [10]. The second technique forces gradual knowledge evaporation, i.e. the particle's best fitness P_{nj}, which initially has been calculated as $J(\mathbf{q}_{nj}^{\text{pbest}})$, is being forgotten exponentially at the rate ρ, bigger than 1 for a positive-definite functional and a minimization task, in the following way [11]:

$$\begin{bmatrix} P_{nj}(i+1) \\ \mathbf{q}_{nj}^{\text{pbest}} \end{bmatrix} = \begin{cases} \begin{bmatrix} \rho P_{nj}(i) \\ \mathbf{q}_{nj}^{\text{pbest}} \end{bmatrix} & \text{if} \quad J(\mathbf{q}_{nj}(i+1)) \geq \rho P_{nj}(i) \\ \begin{bmatrix} J(\mathbf{q}_{nj}(i+1)) \\ \mathbf{q}_{nj}(i+1) \end{bmatrix} & \text{if} \quad J(\mathbf{q}_{nj}(i+1)) < \rho P_{nj}(i) \end{cases}. \tag{5}$$

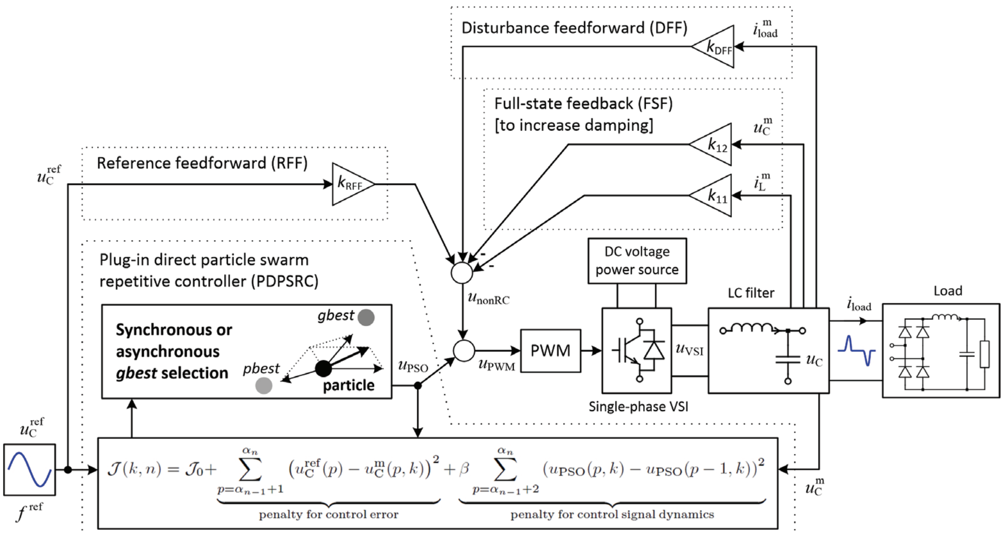

Fig. 1. Plug-in direct particle swarm repetitive controller with accompanying non-repetitive FSF+DFF+RFF controller connected to the CACF VSI loaded by an exemplary diode rectifier

The dynamics of the swarm controller comes directly from the dynamics of the optimizer. Thus, one more key mechanism that influences exploration and exploitation capabilities of the optimizer is scrutinized in this paper, namely the method of determining the *g*best ($\mathbf{q}_n^{\text{gbest}}$). The most common approach is to calculate the gbest as the best of all *p*bests in the synchronous manner [12]. The synchronous update rule (SUR) calculates the new gbest once per swarm iteration at the end of the swarm iteration, i.e. after updating all pbests. A pseudocode for PDPSRC with SUR is as follows:

```
while system is in operation
| for j from 1 to number of particles in subswarm
| | concatenate j-th particles from all subswarms
| | apply the resulting control signal to the plant
| | calculate current fitness in each subswarm
| | update pbest in each subswarm respecting evaporation
| end
| update gbest for each subswarm
| update particles' velocities respecting diversity
| update particles' positions, i.e. control signal
end.
```

Therefore the *g*best as well as velocities and positions are updated outside the inner loop. This means that all particles are updated at the same stage of the algorithm and hence the name "synchronous". In contrast, the asynchronous update rule (ASUR) updates the *g*best just after each pbest update inside the inner loop and its pseudocode is as follows:

```
while system is in operation
| for j from 1 to number of particles in subswarm
| | concatenate j-th particles from all subswarms
| | apply the resulting control signal to the plant
| | calculate current fitness in each subswarm
| | update pbest in each subswarm respecting evaporation
| | update gbest for each subswarm
| | update j-th particles' velocities respecting diversity
| | update j-th particles' positions, i.e. control signal
| end
end.
```

In the ASUR the news about potentially more optimal repetitive control signal are spreading faster. This happens due to the gbest being updated after each period of the reference signal whereas in the SUR the relevant communication takes place only once per S periods of the reference signal, where S denotes the number of particles in a subswarm (identical for all subswarms). The ASUR is often reported to manifest faster convergence but this happens usually at the cost of exploration and can lead to more suboptimal search effects. The No Free Lunch theorem for optimization postulates that the behavior of each optimizer is always problem specific and therefore effectiveness of ASUR versus SUR in PDPSRC can be assessed only in respect to a specific DOP and it should be emphasized that conclusions drawn from such a comparison are not generalizable but nevertheless in some cases may prove to be transferable to PDPSRCs for other plants.

3. Non-repetitive paths

Plug-in repetitive controllers of ILC type, which is the case here, have to be assisted by a non-repetitive controller if there is a need to shape dynamics of a system also in transient states along the pass. This is due to the inherent delay present in the k-direction – the current control error will not be used to correct the control signal until the next pass. The proposed control scheme includes a full-state feedback (FSF), a disturbance static feedforward (DFF) and a reference feedforward (RFF) path to shape the transient response of the CACF VSI in the p-direction. The FSF gains are designed using the pole placement method and its objective is to increase damping in the system that is naturally highly underdamped (see the actual filter resistance and the critical damping resistance for this particular experiment scenario in Tab. 1). This gives a non-RC part of the form

$$u_{\mathrm{nonRC}} = \underbrace{-(k_{11} i_{\mathrm{L}}^{\mathrm{m}} + k_{12} u_{\mathrm{C}}^{\mathrm{m}})}_{\mathrm{FSF}} + \underbrace{(1 + k_{12}) u_{\mathrm{C}}^{\mathrm{ref}}}_{\mathrm{RFF}} + \underbrace{(\hat{R}_{\mathrm{f}} + k_{11}) i_{\mathrm{load}}^{\mathrm{m}}}_{\mathrm{DFF}}, \tag{6}$$

where $\hat{R}_{\mathrm{f}}$ is the identified resistance of the output filter and $\{k_{11}, k_{12}\}$ are FSF controller gains as depicted in Figure 1.

Table 1. Selected parameters of the system

Parameter	**Symbol across the paper**	**Value**
Filter inductance	–	300 μH
Filter capacitance	–	160 μF
Filter resistance	–	0.4 Ω
Critical damping resistance	–	2.74 Ω
Sampling frequency	–	10 kHz
Identified filter resistance	$\hat{R}_{\mathrm{f}}$	0.2 Ω (to make errors more realistic)
Measurement noise	–	depending on the case scenario, 1% or 2% of 100 A or 325 V (unfiltered band-limited white noise with 98% of samples within the range)
Reference frequency	f^{ref}	50 Hz
Pass length	α, i.e. $p \in [1, \alpha]$	200
Number of particles	S	depending on the case scenario, 25 or 10
Swarms' update frequency	f^{ref} / S	depending on the case scenario, 2 Hz or 5 Hz
Number of subswarms	–	5
Points of division (see the cost function (2) with $\alpha_0 = 0$)	α_n	$\alpha_1 = 40, \alpha_2 = 80, \alpha_3 = 120, \alpha_4 = 160$, and $\alpha_5 = \alpha = 200$
Evaporation constant	ρ	1.1
Diversity threshold	D_{thold}	2.0
Penalty factor	β	0.25
Constant summand in (2)	J_0	0.01
Velocity clamping level in (4)	v_{clmp}	9.0

4. Numerical experiment results

Both update strategies (SUR and ASUR) have been tested in exactly the same load conditions to make results comparable. The presupposed scenario is as follows:

a) the swarms are initialized with near zero $\mathbf{u}_{\mathrm{PSO}}(k=0)$ control vector (see Fig. 1), i.e. the particles are randomized within the range of ca. 1% of the maximum control signal amplitude;
b) the resistive load of ca. 4 kW is applied for 100 s or 40 s, respectively for 25 or 10 particles;
c) the resistive load is switched off and the diode rectifier of ca. 6 kW and ca. 2.5 crest factor is switched on for 100 s or 40 s, respectively for 25 or 10 particles;
d) the diode rectifier is switched off and the initial resistive load is applied once again.

The main goal here is to assess and compare both update rules in terms of their rate of convergence, immunity to a given level of measurement noise, and effectiveness in the case of significantly reduced number of particles. Two different noise levels are taken into account: 1% and 2% as described in Table 1. It should be highlighted that this noise is left unfiltered. It is also possible to implement non-causal zero-phase-shift low-pass filtering as the measurement signal conditioning algorithm – a common practice in the classic ILC schemes. However, it has been identified that the proposed PDPSRC is able to operate effectively in the presence of a moderate noise without any filtering. The optimization procedure serves somewhat as the filter thanks to the penalty for control signal dynamics included in the cost functional (2). Any additional low-pass filtering would naturally reduce the controller bandwidth and limit the ability to reject disturbance. The quantitative comparison strategy adopted here uses AUC of the RMSE graphs as a conclusive indicator. This indicator employs control error averaging over the entire period of the reference signal. It is then necessary to scrutinize performance of the system using 2D graphs for transient states and 1D voltage vs. time graphs for steady states. For the sake of compactness, the instantaneous voltage waveforms are displayed only for selected numerical experiments.

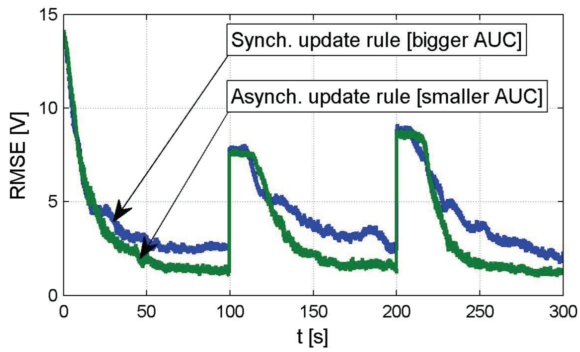

Fig. 2. Evolution of RMSE (calculated over entire period of the reference signal) for swarms with SUR and ASUR – the case scenario of 25 particles and the unfiltered measurement noise equal to 1%

First, the previously developed controller with 25 particles and SUR is compared to its counterpart with ASUR. It can be observed in Figure 2 that AUC of the RMSE graph is clearly smaller in the case of ASUR. The evolution of the output voltage waveform after applying the diode rectifier load is shown in Figure 3. The steady state in the *k*-direction under the same diode rectifier load is illustrated in Figure 4.

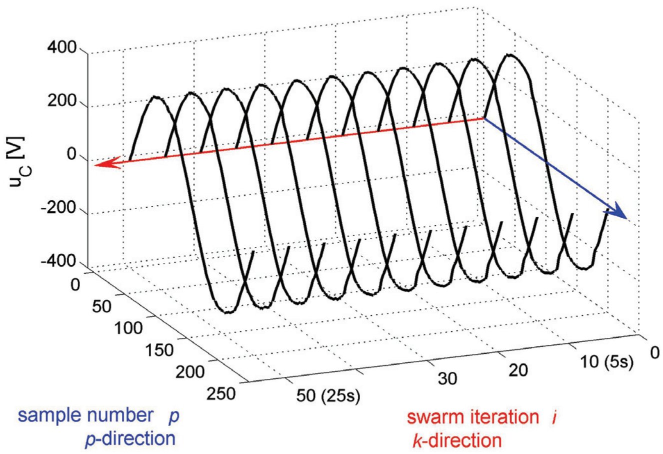

Fig. 3. Evolution of the output voltage waveform after applying the diode rectifier load (in the ASUR case of 25 particles and the unfiltered measurement noise equal to 1%)

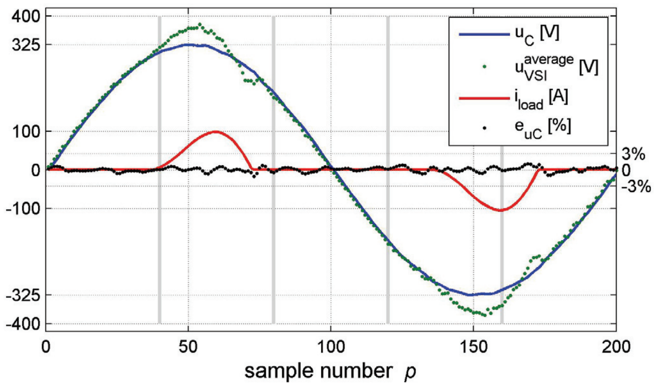

Fig. 4. Quality of the output voltage under the diode rectifier load (steady state in the *k*-direction for PDPSRC with ASUR for 25 particles and the unfiltered measurement noise equal to 1%)

The observed faster convergence and more consistent behavior of the swarm should be attributed to increased knowledge dissemination between particles in the asynchronous method. By consistent it is meant here that the swarm tends to move towards a new minimum "more" monotonically.

Next, the robustness to noise is tested. The swarm updated synchronously is incomparably less immune to noise than the asynchronous one (see Fig. 5). This could be associated with better noise handling by averaging during more frequent gbest updates in ASUR.

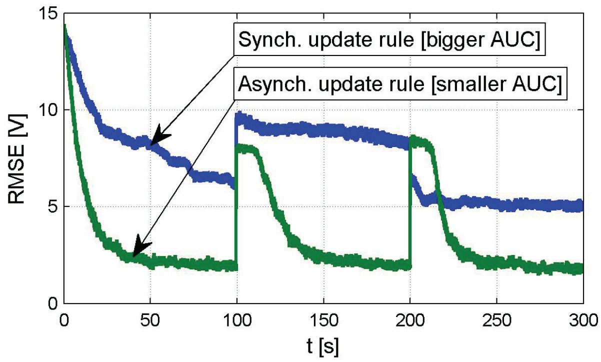

Fig. 5. Evolution of RMSE (calculated over entire period of the reference signal) for swarms with SUR and ASUR – the case scenario of 25 particles and the unfiltered measurement noise equal to 2%

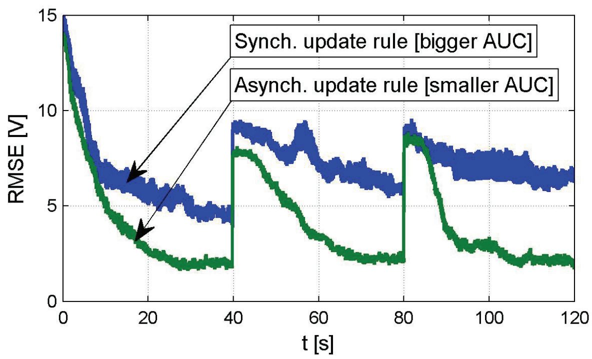

Fig. 6. Evolution of RMSE (calculated over entire period of the reference signal) for swarms with SUR and ASUR – the case scenario of 10 particles and the unfiltered measurement noise equal to 1%

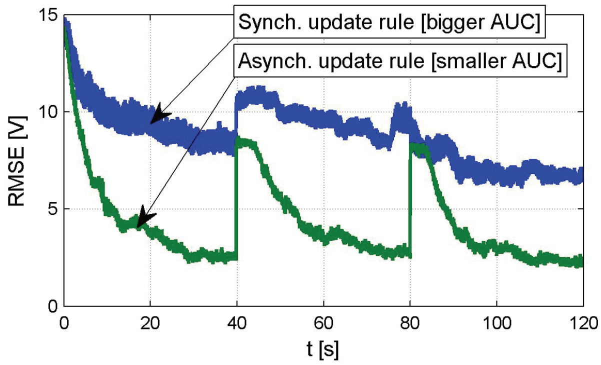

Fig. 7. Evolution of RMSE (calculated over entire period of the reference signal) for swarms with SUR and ASUR – the case scenario of 10 particles and the unfiltered measurement noise equal to 2%

Finally, a reduction in number of particles is investigated. In off-line (non-real-time) optimization problems, usually 30÷50 particles are used as a trade-off between the accuracy of a search and the wall-clock time needed to complete it. Similar compromise has to be worked out also in on-line optimization problems. The DOP iterative solvers that act as closed-loop real-time controllers are particularly challenging in this regard. In the discussed PDPSRC the number of particles should be kept as small as possible. The size of the swarm directly affects responsiveness of the controller. One period of the reference voltage is needed to rate one particle. Therefore, to rate, e.g., 25 particles exactly 25 periods of the reference signal are required and the resulting swarm update frequency is 2 Hz for 50 Hz of the desired output voltage frequency. It is then clear that smaller swarms can be updated more frequently and potentially better responsiveness, i.e. higher dynamics in the *k*-direction, could be achieved. It should be noted that splitting swarms into adjacent subswarms does not affect the update rate. Although smaller swarms can be updated more frequently, they also often manifest poor convergence on rugged landscapes. Figures 6 and 7 demonstrate that the ASUR is better candidate for PDPSRC than the SUR if smaller swarms are to be used. The 10-particle swarm is unable to operate effectively if governed by the SUR whereas 10 particles updated asynchronously are capable of tracking the moving optimum. The dynamics in the *k*-direction has been then noticeably improved (Figs. 6 and 7 vs. Figs. 2 and 5) at the cost of slightly more distorted output voltage waveform (Fig. 9 vs. Fig. 4). A selected transient state in the case of the 10-particle swarm is shown in Figure 8.

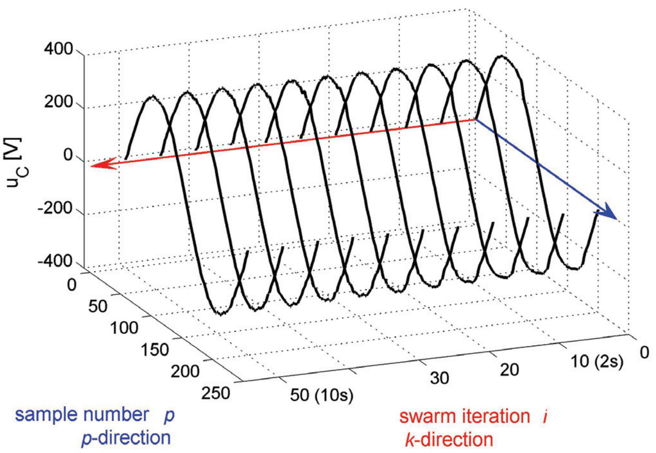

Fig. 8. Evolution of the output voltage waveform after applying the diode rectifier load (in the ASUR case of 10 particles and the unfiltered measurement noise equal to 1%)

It should be stressed that all comparisons carried in this study are valid only for the system tuned as in Table 1. The PDPSRC has several parameters that have to be set – currently by guessing and checking – and majority of them have significant influence on the dynamics of the RC. At the moment no systematic comparison has been carried out between SUR and ASUR in relation to the proposed RC under different controller settings. It is planned that such a comparison will be the subject of future studies and should include not only various levels of the measurement noise and different sizes of the swarm but also different settings for the

DOP-capable PSO, namely for the evaporation constant, the diversity threshold as well as for the velocity clamping level.

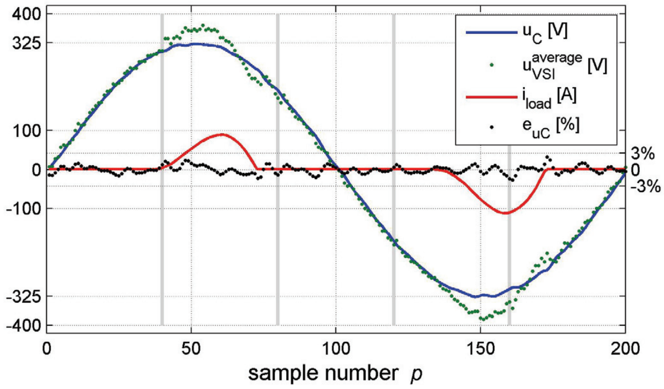

Fig. 9. Quality of the output voltage under the diode rectifier load (steady state in the *k*-direction for PDPSRC with ASUR for 10 particles and the unfiltered measurement noise equal to 1%)

5. Conclusions

A repetitive controller for the constant-amplitude constant-frequency inverter has been presented. The control task in the repetitive direction has been formulated in such a way that it poses a dynamic optimization problem. This problem is being then solved in real time using a particle swarm based optimizer. Naturally, the basic swarm update rule has been modified to cope with non-stationarity of the process at hand that arises from unpredictable variable load conditions. Therefore, the diversity supervision and knowledge gradual forgetting mechanisms have been embedded into the optimizer. Then, two alternative algorithm workflows, namely synchronous and asynchronous population evolution, have been investigated. The swarm governed by the asynchronous update rule manifests smaller area under the root mean squared control error. Obviously, this conclusion is drawn under the particular case scenario described and for the certain controller settings given. Results of comparisons between asynchronous and synchronous swarms suggest improved efficacy of the former in coping with noise and small size swarm circumstance. The direct advantage of the asynchronous rule is then more effective dealing with the changing environment and measurement uncertainties. Further study is planned to determine whether the superiority of the asynchronous population evolution in the repetitive controller is definitive. Nevertheless, it can be already concluded that the asynchronous update rule exhibits features desirable for the real-time iterative learning systems such as the developed plug-in direct particle swarm repetitive controller.

Acknowledgement
This research was partially supported by the statutory fund of Electrical Drive Division within the Institute of Control and Industrial Electronics (2014).

References

[1] Ufnalski B., Grzesiak L.M., Kaszewski A., *Advanced control and optimization techniques in AC drives and DC/AC sine wave voltage inverters: selected problems.* Advanced and intelligent control in power electronics and drives, Springer (2014).

[2] Ufnalski B., Grzesiak L.M., Galkowski K., *Particle swarm optimization of an iterative learning controller for the single-phase inverter with sinusoidal output voltage waveform.* Bulletin of the Polish Academy of Sciences – Technical Sciences 61(3): 649-660 (2013).

[3] Longman R.W., *Iterative/repetitive learning control: learning from theory, simulations, and experiments.* Encyclopedia of the Sciences of Learning, pp. 1652-1657, Springer US (2012).

[4] Shi Y., *Robustification in repetitive and iterative learning control.* Ph.D. thesis, Columbia University, USA (2013).

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

[6] Ufnalski B., Grzesiak L.M., *Artificial neural network based voltage controller for the single phase true sine wave inverter – a repetitive control approach.* Electrical Review (Przegląd Elektrotechniczny), in English, 84(4): 14-18 (2013).

[7] Ufnalski B., Grzesiak L.M., *Particle swarm optimization of an online trained repetitive neurocontroller fot the sine-wave inverter.* Proc. of IEEE Industrial Electronics Society Annual Conference IECON 2013, pp. 6001-6007 (2013).

[8] Ufnalski B., Grzesiak L.M., *A plug-in direct particle swarm repetitive controller for a single-phase inverter.* Electrical Review 90(6): 6-11 (2014) (in English).

[9] Ufnalski B., Grzesiak L.M., *Feedback and feedforward repetitive control of single-phase UPS inverters – an online particle swarm optimization approach.* Scientific Reports of the Cologne University of Applied Sciences 1: 59-67 (2014).

[10] Riget J., Vesterstrom J.S., *A diversity-guided particle swarm optimizer – the ARPSO.* Tech. Rep. of Aarhus University, Denmark (2002).

[11] Cui X., Charles J.S., Potok T.E., *A simple distributed particle swarm optimization for dynamic and noisy environments.* Nature Inspired Cooperative Strategies for Optimization (NICSO 2008), Studies in Computational Intelligence 236: 89-102, Springer (2009).

[12] Rada-Vilela J., Zhang M., Seah W., *A performance study on synchronous and asynchronous updates in particle swarm optimization.* Proc. of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO'11), pp. 21-28 (2011).