



WIKIPEDIA
The Free Encyclopedia

[Main page](#)
[Contents](#)
[Featured content](#)
[Current events](#)
[Random article](#)
[Donate to Wikipedia](#)
[Wikipedia store](#)

Interaction

[Help](#)
[About Wikipedia](#)
[Community portal](#)
[Recent changes](#)
[Contact page](#)

Tools

[What links here](#)
[Related changes](#)
[Upload file](#)
[Special pages](#)
[Permanent link](#)
[Page information](#)
[Wikidata item](#)
[Cite this page](#)

Print/export

[Create a book](#)
[Download as PDF](#)
[Printable version](#)

Languages

[العربية](#)
[Čeština](#)
[Deutsch](#)
[Español](#)
[فارسی](#)
[Français](#)
[हिन्दी](#)
[Italiano](#)
[עברית](#)
[日本語](#)
[Português](#)
[Русский](#)
[Українська](#)
[Tiếng Việt](#)
[中文](#)

 [Edit links](#)

[Create account](#) [Log in](#)

Article [Talk](#)

[Read](#) [Edit](#) [View history](#)

Particle swarm optimization

From Wikipedia, the free encyclopedia

In [computer science](#), **particle swarm optimization (PSO)** is a computational method that [optimizes](#) a problem by [iteratively](#) trying to improve a [candidate solution](#) with regard to a given measure of quality. PSO optimizes a problem by having a population of candidate solutions, here dubbed [particles](#), and moving these particles around in the [search-space](#) according to simple [mathematical formulae](#) over the particle's [position](#) and [velocity](#). Each particle's movement is influenced by its local best known position but, is also guided toward the best known positions in the search-space, which are updated as better positions are found by other particles. This is expected to move the swarm toward the best solutions.

PSO is originally attributed to [Kennedy](#), [Eberhart](#) and Shi^{[1][2]} and was first intended for [simulating social behaviour](#),^[3] as a stylized representation of the movement of organisms in a bird [flock](#) or [fish school](#). The algorithm was simplified and it was observed to be performing optimization. The book by Kennedy and Eberhart^[4] describes many philosophical aspects of PSO and [swarm intelligence](#). An extensive survey of PSO applications is made by [Poli](#).^{[5][6]}

PSO is a [metaheuristic](#) as it makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. However, metaheuristics such as PSO do not guarantee an optimal solution is ever found. More specifically, PSO does not use the [gradient](#) of the problem being optimized, which means PSO does not require that the optimization problem be [differentiable](#) as is required by classic optimization methods such as [gradient descent](#) and [quasi-newton methods](#). PSO can therefore also be used on optimization problems that are partially irregular, noisy, change over time, etc.

Contents [\[hide\]](#)

- Algorithm
- Parameter selection
- Neighborhoods and Topologies
- Inner workings
 - Convergence
 - Biases
- Variants
 - Hybridization
 - Alleviate Premature
 - Simplifications
 - Multi-objective optimization
 - Binary, Discrete, and Combinatorial PSO
- Applications
 - Optimization of hydrocarbon field development planning
- See also
- References
- External links

Algorithm [\[edit\]](#)

A basic variant of the PSO algorithm works by having a population (called a swarm) of [candidate solutions](#) (called particles). These particles are moved around in the search-space according to a few simple formulae. The movements of the particles are guided by their own best known position in the search-space as well as the entire swarm's best known position. When improved positions are being discovered these will then come to guide the movements of the swarm. The process is repeated and by doing so it is hoped, but not guaranteed, that a satisfactory solution will eventually be discovered.

Formally, let $f: \mathcal{I}^n \rightarrow \mathbb{I}$ be the cost function which must be minimized. The function takes a candidate solution as argument in the form of a [vector](#) of [real numbers](#) and produces a real number as output which indicates the objective function value of the given candidate solution. The [gradient](#) of f is not known. The goal is to find a solution \mathbf{a} for which $f(\mathbf{a}) \leq f(\mathbf{b})$ for all \mathbf{b} in the search-space, which would mean \mathbf{a} is the global minimum. Maximization can be performed by considering the function $h = -f$ instead.

Let S be the number of particles in the swarm, each having a position $\mathbf{x}_i \in \mathbb{I}^n$ in the search-space and a velocity $\mathbf{v}_i \in \mathbb{I}^n$. Let \mathbf{p}_i be the best known position of particle i and let \mathbf{g} be the best known position of the entire swarm. A basic PSO algorithm is then:^[7]

- For each particle $i = 1, \dots, S$ do:
 - Initialize the particle's position with a [uniformly distributed](#) random vector: $\mathbf{x}_i \sim U(\mathbf{b}_{\text{lo}}, \mathbf{b}_{\text{up}})$, where \mathbf{b}_{lo} and \mathbf{b}_{up} are the lower and upper boundaries of the search-space.
 - Initialize the particle's best known position to its initial position: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
 - If $(f(\mathbf{p}_i) < f(\mathbf{g}))$ update the swarm's best known position: $\mathbf{g} \leftarrow \mathbf{p}_i$
 - Initialize the particle's velocity: $\mathbf{v}_i \sim U(-|\mathbf{b}_{\text{up}} - \mathbf{b}_{\text{lo}}|, |\mathbf{b}_{\text{up}} - \mathbf{b}_{\text{lo}}|)$
- Until a termination criterion is met (e.g. number of iterations performed, or a solution with adequate objective function value is found), repeat:
 - For each particle $i = 1, \dots, S$ do:
 - For each dimension $d = 1, \dots, n$ do:
 - Pick random numbers: $r_p, r_g \sim U(0, 1)$
 - Update the particle's velocity: $\mathbf{v}_{i,d} \leftarrow \omega \mathbf{v}_{i,d} + \phi_p r_p (\mathbf{p}_{i,d} - \mathbf{x}_{i,d}) + \phi_g r_g (\mathbf{g}_d - \mathbf{x}_{i,d})$
 - Update the particle's position: $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
 - If $(f(\mathbf{x}_i) < f(\mathbf{p}_i))$ do:
 - Update the particle's best known position: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
 - If $(f(\mathbf{p}_i) < f(\mathbf{g}))$ update the swarm's best known position: $\mathbf{g} \leftarrow \mathbf{p}_i$
 - Now \mathbf{g} holds the best found solution.

The parameters ω , ϕ_p , and ϕ_g are selected by the practitioner and control the behaviour and efficacy of the PSO method, see [below](#).

Parameter selection [\[edit\]](#)

The choice of PSO parameters can have a large impact on optimization performance. Selecting PSO parameters that yield good performance has therefore been the subject of much research.^{[8][9][10][11][12][13][14][15]}

The PSO parameters can also be tuned by using another overlaying optimizer, a concept known as [meta-optimization](#).^{[16][17][18]} Parameters have also been tuned for various optimization scenarios.^[19]

Neighborhoods and Topologies [\[edit\]](#)

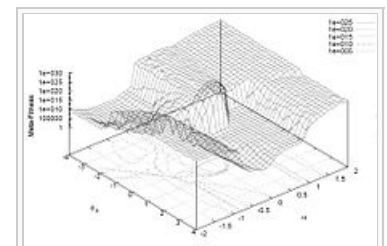
The basic PSO is easily trapped into a local minimum. This premature convergence can be avoided by not using the entire swarm's best known position \mathbf{g} but just the best known position \mathbf{l} of a sub-swarm "around" the particle that is moved. Such a sub-swarm can be a geometrical one - for example "the m nearest particles" - or, more often, a social one, i.e. a set of particles that is not depending on any distance. In such a case, the PSO variant is said to be local best (vs global best for the basic PSO).

If we suppose there is an information link between each particle and its neighbours, the set of these links builds a graph, a communication network, that is called the topology of the PSO variant. A commonly used social topology is the ring, in which each particle has just two neighbours, but there are many others.^[20] The topology is not necessarily fixed, and can be adaptive (SPSO,^[21] stochastic star,^[22] TRIBES,^[23] Cyber Swarm,^[24] C-PSO^[25]).

Inner workings [\[edit\]](#)

There are several [schools of thought](#) as to why and how the PSO algorithm can perform optimization.

A common belief amongst researchers is that the swarm behaviour varies between exploratory behaviour, that is, searching a broader region of the search-space, and exploitative behaviour, that is, a locally oriented search so as to get closer to a (possibly local) optimum. This school of thought has been prevalent since the inception of PSO.^{[2][3][8][12]} This school of thought contends that the PSO algorithm and its parameters must be chosen so as to properly balance between exploration and exploitation to avoid [premature convergence](#) to a [local optimum](#) yet still ensure a good rate of [convergence](#) to the optimum. This belief is the precursor of many PSO variants, see [below](#).



Performance landscape showing how a simple PSO variant performs in aggregate on several benchmark problems when varying two PSO parameters.

Another school of thought is that the behaviour of a PSO swarm is not well understood in terms of how it affects actual optimization performance, especially for higher-dimensional search-spaces and optimization problems that may be discontinuous, noisy, and time-varying. This school of thought merely tries to find PSO algorithms and parameters that cause good performance regardless of how the swarm behaviour can be interpreted in relation to e.g. exploration and exploitation. Such studies have led to the simplification of the PSO algorithm, see [below](#).

Convergence [\[edit\]](#)

In relation to PSO the word *convergence* typically refers to two different definitions:

- Convergence of the sequence of solutions (aka, stability analysis, [converging](#)) in which all particles have converged to a point in the search-space, which may or may not be the optimum,
- Convergence to a local optimum where all personal bests **p** or, alternatively, the swarm's best known position **g**, approaches a local optimum of the problem, regardless of how the swarm behaves.

Convergence of the sequence of solutions has been investigated for PSO.^{[11][12][13]} These analyses have resulted in guidelines for selecting PSO parameters that are believed to cause convergence to a point and prevent divergence of the swarm's particles (particles do not move unboundedly and will converge to somewhere). However, the analyses were criticized by Pedersen^[18] for being oversimplified as they assume the swarm has only one particle, that it does not use stochastic variables and that the points of attraction, that is, the particle's best known position **p** and the swarm's best known position **g**, remain constant throughout the optimization process. However, it was shown ^[26] that these simplifications do not affect the boundaries found by these studies for parameter where the swarm is convergent.

Convergence to a local optimum has been analyzed for PSO in ^[27] and ^[28]. It has been proven that PSO need some modification to guarantee to find a local optimum.

This means that determining convergence capabilities of different PSO algorithms and parameters therefore still depends on [empirical](#) results. One attempt at addressing this issue is the development of an "orthogonal learning" strategy for an improved use of the information already existing in the relationship between **p** and **g**, so as to form a leading converging exemplar and to be effective with any PSO topology. The aims are to improve the performance of PSO overall, including faster global convergence, higher solution quality, and stronger robustness.^[29] However, such studies do not provide theoretical evidence to actually prove their claims.

Biases [\[edit\]](#)

As the basic PSO works dimension by dimension, the solution point is easier found when it lies on an axis of the search space, on a diagonal, and even easier if it is right on the centre.^{[30][31]}

One approach is to modify the algorithm so that it is not any more sensitive to the system of coordinates.^{[32][33][34][35]} Note that some of these methods have a higher computational complexity (are in $O(n^2)$ where n is the number of dimensions) that make the algorithm very slow for large scale optimization.^[28]

The only currently existing PSO variant that is not sensitive to the rotation of the coordinates while is locally convergent has been proposed at 2014.^[28] The method has shown a very good performance on many benchmark problems while its rotation invariance and local convergence have been mathematically proven.

Variants [\[edit\]](#)

Numerous variants of even a basic PSO algorithm are possible. For example, there are different ways to initialize the particles and velocities (e.g. start with zero velocities instead), how to dampen the velocity, only update **p_i** and **g** after the entire swarm has been updated, etc. Some of these choices and their possible performance impact have been discussed in the literature.^[10]

Hybridization [\[edit\]](#)

New and more sophisticated PSO variants are also continually being introduced in an attempt to improve optimization performance. There are certain trends in that research; one is to make a hybrid optimization method using PSO combined with other optimizers,^{[36][37][38]} e.g., combined PSO with biogeography-based optimization,^[39] and the incorporation of an effective learning method.^[29]

Alleviate Premature [\[edit\]](#)

Another research trend is to try and alleviate premature convergence (that is, optimization stagnation), e.g. by reversing or perturbing the movement of the PSO particles,^{[15][40][41][42]} another approach to deal with

premature convergence is the use of multiple swarms^[43] ([multi-swarm optimization](#)). The multi-swarm approach can also be used to implement multi-objective optimization.^[44] Finally, there are developments in adapting the behavioural parameters of PSO during optimization.^[45]

Simplifications [\[edit\]](#)

Another school of thought is that PSO should be simplified as much as possible without impairing its performance; a general concept often referred to as [Occam's razor](#). Simplifying PSO was originally suggested by Kennedy^[3] and has been studied more extensively,^{[14][17][18][46]} where it appeared that optimization performance was improved, and the parameters were easier to tune and they performed more consistently across different optimization problems.

Another argument in favour of simplifying PSO is that [metaheuristics](#) can only have their efficacy demonstrated [empirically](#) by doing computational experiments on a finite number of optimization problems. This means a metaheuristic such as PSO cannot be [proven correct](#) and this increases the risk of making errors in its description and implementation. A good example of this^[47] presented a promising variant of a [genetic algorithm](#) (another popular metaheuristic) but it was later found to be defective as it was strongly biased in its optimization search towards similar values for different dimensions in the search space, which happened to be the optimum of the benchmark problems considered. This bias was because of a programming error, and has now been fixed.^[48]

Initialization of velocities may require extra inputs. A simpler variant is the accelerated particle swarm optimization (APSO),^[49] which does not need to use velocity at all and can speed up the convergence in many applications. A simple demo code of APSO is available.^[50]

Multi-objective optimization [\[edit\]](#)

PSO has also been applied to [multi-objective problems](#),^{[51][52]} in which the objective function comparison takes [pareto dominance](#) into account when moving the PSO particles and non-dominated solutions are stored so as to approximate the pareto front.

Binary, Discrete, and Combinatorial PSO [\[edit\]](#)

As the PSO equations given above work on real numbers, a commonly used method to solve discrete problems is to map the discrete search space to a continuous domain, to apply a classical PSO, and then to demap the result. Such a mapping can be very simple (for example by just using rounded values) or more sophisticated.^[53]

However, it can be noted that the equations of movement make use of operators that perform four actions:

- computing the difference of two positions. The result is a velocity (more precisely a displacement)
- multiplying a velocity by a numerical coefficient
- adding two velocities
- applying a velocity to a position

Usually a position and a velocity are represented by n real numbers, and these operators are simply $-$, $*$, $+$, and again $+$. But all these mathematical objects can be defined in a completely different way, in order to cope with binary problems (or more generally discrete ones), or even combinatorial ones.^{[54][55][56][57]} One approach is to redefine the operators based on sets.^[58]

Applications [\[edit\]](#)

























Optimization of hydrocarbon field development planning [\[edit\]](#)




Many of the optimization problems in science and engineering involve nonlinear objective functions. Particle swarm optimization has been applied for optimization of oil and gas field development planning.^[59]

See also [\[edit\]](#)

- [Swarm intelligence](#)
- [Multi-swarm optimization](#)
- [Bees algorithm](#) / [Artificial Bee Colony Algorithm](#)
- [Particle filter](#)
- [Derivative-free optimization](#)

References [\[edit\]](#)

1. ^a Kennedy, J.; Eberhart, R. (1995). "Particle Swarm Optimization" . *Proceedings of IEEE International Conference on Neural Networks IV*. pp. 1942–1948. doi:10.1109/ICNN.1995.488968 .
2. ^{a b} Shi, Y.; Eberhart, R.C. (1998). "A modified particle swarm optimizer". *Proceedings of IEEE International Conference on Evolutionary Computation*. pp. 69–73.
3. ^{a b c} Kennedy, J. (1997). "The particle swarm: social adaptation of knowledge". *Proceedings of IEEE International Conference on Evolutionary Computation*. pp. 303–308.
4. ^a Kennedy, J.; Eberhart, R.C. (2001). *Swarm Intelligence*. Morgan Kaufmann. ISBN 1-55860-595-9.
5. ^a Poli, R. (2007). "An analysis of publications on particle swarm optimisation applications"  (PDF). *Technical Report CSM-469* (Department of Computer Science, University of Essex, UK).
6. ^a Poli, R. (2008). "Analysis of the publications on the applications of particle swarm optimisation"  (PDF). *Journal of Artificial Evolution and Applications* **2008**: 1–10. doi:10.1155/2008/685175 .
7. ^a Clerc, M. (2012). "Standard Particle Swarm Optimisation"  (PDF). *HAL open access archive*.
8. ^{a b} Shi, Y.; Eberhart, R.C. (1998). "Parameter selection in particle swarm optimization". *Proceedings of Evolutionary Programming VII (EP98)*. pp. 591–600.
9. ^a Eberhart, R.C.; Shi, Y. (2000). "Comparing inertia weights and constriction factors in particle swarm optimization". *Proceedings of the Congress on Evolutionary Computation* **1**. pp. 84–88.
10. ^{a b} Carlisle, A.; Dozier, G. (2001). "An Off-The-Shelf PSO"  (PDF). *Proceedings of the Particle Swarm Optimization Workshop*. pp. 1–6.
11. ^{a b} van den Bergh, F. (2001). *An Analysis of Particle Swarm Optimizers* (PhD thesis). University of Pretoria, Faculty of Natural and Agricultural Science.
12. ^{a b c} Clerc, M.; Kennedy, J. (2002). "The particle swarm - explosion, stability, and convergence in a multidimensional complex space". *IEEE Transactions on Evolutionary Computation* **6** (1): 58–73. doi:10.1109/4235.985692 .
13. ^{a b} Trelea, I.C. (2003). "The Particle Swarm Optimization Algorithm: convergence analysis and parameter selection". *Information Processing Letters* **85** (6): 317–325. doi:10.1016/S0020-0190(02)00447-7 .
14. ^{a b} Bratton, D.; Blackwell, T. (2008). "A Simplified Recombinant PSO". *Journal of Artificial Evolution and Applications*.
15. ^{a b} Evers, G. (2009). *An Automatic Regrouping Mechanism to Deal with Stagnation in Particle Swarm Optimization*  (Master's thesis). The University of Texas - Pan American, Department of Electrical Engineering.
16. ^a Meissner, M.; Schmuker, M.; Schneider, G. (2006). "Optimized Particle Swarm Optimization (OPSO) and its application to artificial neural network training" . *BMC Bioinformatics* **7** (1): 125. doi:10.1186/1471-2105-7-125 . PMC 1464136 . PMID 16529661 .
17. ^{a b} Pedersen, M.E.H. (2010). *Tuning & Simplifying Heuristical Optimization*  (PhD thesis). University of Southampton, School of Engineering Sciences, Computational Engineering and Design Group.
18. ^{a b c} Pedersen, M.E.H.; Chipperfield, A.J. (2010). "Simplifying particle swarm optimization"  (PDF). *Applied Soft Computing* **10** (2): 618–628. doi:10.1016/j.asoc.2009.08.029 .
19. ^a Pedersen, M.E.H. (2010). "Good parameters for particle swarm optimization"  (PDF). *Technical Report HL1001* (Hass Laboratories).
20. ^a Mendes, R. (2004). *Population Topologies and Their Influence in Particle Swarm Performance* (PhD thesis). Universidade do Minho.
21. ^a SPSO, [Particle Swarm Central](#) .
22. ^a Miranda, V., Keko, H. and Duque, Á. J. (2008). Stochastic Star Communication Topology in Evolutionary Particle Swarms (EPSO). *International Journal of Computational Intelligence Research (IJCIR)*, Volume 4, Number 2, pp. 105-116
23. ^a Clerc, M. (2006). Particle Swarm Optimization. ISTE (International Scientific and Technical Encyclopedia), 2006
24. ^a Yin, P., Glover, F., Laguna, M., & Zhu, J. (2011). A Complementary Cyber Swarm Algorithm. *International Journal of Swarm Intelligence Research (IJSIR)*, 2(2), 22-41
25. ^a Elshamy, W.; Rashad, H.; Bahgat, A. (2007). "Clubs-based Particle Swarm Optimization"  (PDF). *IEEE Swarm Intelligence Symposium 2007 (SIS2007)*. Honolulu, HI. pp. 289–296.
26. ^a Cleghorn, Christopher W (2014). "Particle Swarm Convergence: Standardized Analysis and Topological Influence". *Swarm Intelligence Conference*.
27. ^a Van den Bergh, F. "A convergence proof for the particle swarm optimiser". *Fundamenta Informaticae*.
28. ^{a b c} Bonyadi, Mohammad reza.; Michalewicz, Z. (2014). "A locally convergent rotationally invariant particle swarm optimization algorithm". *Swarm intelligence* **8** (3): 159–198. doi:10.1007/s11721-014-0095-1 .
29. ^{a b} Zhan, Z-H.; Zhang, J.; Li, Y.; Shi, Y-H. (2011). "Orthogonal Learning Particle Swarm Optimization"  (PDF). *IEEE Transactions on Evolutionary Computation* **15** (6): 832–847. doi:10.1109/TEVC.2010.2052054 .
30. ^a Monson, C. K. & Seppi, K. D. (2005). Exposing Origin-Seeking Bias in PSO GECCO'05, pp. 241-248
31. ^a Spears, W. M., Green, D. T. & Spears, D. F. (2010). Biases in Particle Swarm Optimization. *International Journal of Swarm Intelligence Research*, Vol. 1(2), pp. 34-57
32. ^a Wilke, D. N., Kok, S. & Groenwold, A. A. (2007). Comparison of linear and classical velocity update rules in particle swarm optimization: notes on scale and frame invariance. *International Journal for Numerical Methods in Engineering*, John Wiley & Sons, Ltd., 70, pp. 985-1008
33. ^a SPSO 2011, [Particle Swarm Central](#) 

34. [▲] Bonyadi, Mohammad reza; Michalewicz, Z. (2014). "SPSO 2011 analysis of stability; local convergence; and rotation sensitivity.". *GECCO2014 (the best paper award in the track ACSI)*: 9–16.
35. [▲] Bonyadi, Mohammad reza.; Michalewicz, Z. (2014). "An analysis of the velocity updating rule of the particle swarm optimization algorithm". *Journal of Heuristics* **20** (4): 417–452. doi:10.1007/s10732-014-9245-2 [↗](#).
36. [▲] Lovbjerg, M.; Krink, T. (2002). "The LifeCycle Model: combining particle swarm optimisation, genetic algorithms and hillclimbers". *Proceedings of Parallel Problem Solving from Nature VII (PPSN)*. pp. 621–630.
37. [▲] Niknam, T.; Amiri, B. (2010). "An efficient hybrid approach based on PSO, ACO and k-means for cluster analysis". *Applied Soft Computing* **10** (1): 183–197. doi:10.1016/j.asoc.2009.07.001 [↗](#).
38. [▲] Zhang, Wen-Jun; Xie, Xiao-Feng (2003). *DEPSO: hybrid particle swarm with differential evolution operator* . *IEEE International Conference on Systems, Man, and Cybernetics (SMCC)*, Washington, DC, USA: 3816-3821.
39. [▲] Zhang, Y.; Wang, S. (2015). "Pathological Brain Detection in Magnetic Resonance Imaging Scanning by Wavelet Entropy and Hybridization of Biogeography-based Optimization and Particle Swarm Optimization" [↗](#). *Progress in Electromagnetics Research – Pier* **152**: 41–58.
40. [▲] Lovbjerg, M.; Krink, T. (2002). "Extending Particle Swarm Optimisers with Self-Organized Criticality". *Proceedings of the Fourth Congress on Evolutionary Computation (CEC)* **2**. pp. 1588–1593.
41. [▲] Xinchao, Z. (2010). "A perturbed particle swarm algorithm for numerical optimization". *Applied Soft Computing* **10** (1): 119–124. doi:10.1016/j.asoc.2009.06.010 [↗](#).
42. [▲] Xie, Xiao-Feng; Zhang, Wen-Jun; Yang, Zhi-Lian (2002). *A dissipative particle swarm optimization* . *Congress on Evolutionary Computation (CEC)*, Honolulu, HI, USA: 1456-1461.
43. [▲] Cheung, N. J., Ding, X-M., & Shen, H-B. (2013). OptiFel: A Convergent Heterogeneous Particle Swarm Optimization Algorithm for Takagi-Sugeno Fuzzy Modeling, *IEEE Transactions on Fuzzy Systems*, doi:10.1109/TFUZZ.2013.2278972 [↗](#)
44. [▲] Nobile, M.; Besozzi, D.; Cazzaniga, P.; Mauri, G.; Pescini, D. (2012). "A GPU-Based Multi-Swarm PSO Method for Parameter Estimation in Stochastic Biological Systems Exploiting Discrete-Time Target Series" [↗](#). *Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics. Lecture Notes in Computer Science*. **7264**. pp. 74–85.
45. [▲] Zhan, Z-H.; Zhang, J.; Li, Y.; Chung, H.S-H. (2009). "Adaptive Particle Swarm Optimization"  (PDF). *IEEE Transactions on Systems, Man, and Cybernetics* **39** (6): 1362–1381. doi:10.1109/TSMCB.2009.2015956 [↗](#).
46. [▲] Yang, X.S. (2008). *Nature-Inspired Metaheuristic Algorithms*. Luniver Press. ISBN 978-1-905986-10-1.
47. [▲] Tu, Z.; Lu, Y. (2004). "A robust stochastic genetic algorithm (StGA) for global numerical optimization". *IEEE Transactions on Evolutionary Computation* **8** (5): 456–470. doi:10.1109/TEVC.2004.831258 [↗](#).
48. [▲] Tu, Z.; Lu, Y. (2008). "Corrections to "A Robust Stochastic Genetic Algorithm (StGA) for Global Numerical Optimization". *IEEE Transactions on Evolutionary Computation* **12** (6): 781–781. doi:10.1109/TEVC.2008.926734 [↗](#).
49. [▲] X. S. Yang, S. Deb and S. Fong, Accelerated particle swarm optimization and support vector machine for business optimization and applications, *NDT 2011, Springer CCIS 136*, pp. 53-66 (2011).
50. [▲] <http://www.mathworks.com/matlabcentral/fileexchange/?term=APSO> [↗](#)
51. [▲] Parsopoulos, K.; Vrahatis, M. (2002). "Particle swarm optimization method in multiobjective problems" [↗](#). *Proceedings of the ACM Symposium on Applied Computing (SAC)*. pp. 603–607.
52. [▲] Coello Coello, C.; Salazar Lechuga, M. (2002). "MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization" [↗](#). *Congress on Evolutionary Computation (CEC'2002)*. pp. 1051–1056.
53. [▲] Roy, R., Dehuri, S., & Cho, S. B. (2012). A Novel Particle Swarm Optimization Algorithm for Multi-Objective Combinatorial Optimization Problem. 'International Journal of Applied Metaheuristic Computing (IJAMC)', 2(4), 41-57
54. [▲] Kennedy, J. & Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm, *Conference on Systems, Man, and Cybernetics*, Piscataway, NJ: IEEE Service Center, pp. 4104-4109
55. [▲] Clerc, M. (2004). *Discrete Particle Swarm Optimization, illustrated by the Traveling Salesman Problem*, *New Optimization Techniques in Engineering*, Springer, pp. 219-239
56. [▲] Clerc, M. (2005). Binary Particle Swarm Optimisers: toolbox, derivations, and mathematical insights, [Open Archive HAL](#) [↗](#)
57. [▲] Jarboui, B., Damak, N., Siarry, P., and Rebai, A.R. (2008). A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems. In *Proceedings of Applied Mathematics and Computation*, pp. 299-308.
58. [▲] Chen, Wei-neng; Zhang, Jun (2010). "A novel set-based particle swarm optimization method for discrete optimization problem". *IEEE Transactions on Evolutionary Computation* **14** (2): 278–300. doi:10.1109/tevc.2009.2030331 [↗](#).
59. [▲] "Closed-loop field development under uncertainty using optimization with sample validation" [↗](#). *SPE Journal*. doi:10.2118/173219-PA [↗](#).

External links [\[edit\]](#)

- [Particle Swarm Central](#) [↗](#) is a repository for information on PSO. Several source codes are freely available.
- [A brief video](#) [↗](#) of particle swarms optimizing three benchmark functions.
- [Applications](#) [↗](#) of PSO.

- [Automatic Calibration of a Rainfall-Runoff Model Using a Fast and Elitist Multi-objective Particle Swarm Algorithm](#)
- [Particle Swarm Optimization](#) (see and listen to Lecture 27)
- [Links to PSO source code](#)

| | | |
|--|--|---------------------|
| v · t · e | Major subfields of optimization | [show] |
| v · t · e | Swarming | [show] |

| |
|---|
| Categories: Optimization algorithms and methods Evolutionary algorithms |
|---|

This page was last modified on 4 September 2015, at 05:48.

Text is available under the [Creative Commons Attribution-ShareAlike License](#); additional terms may apply. By using this site, you agree to the [Terms of Use](#) and [Privacy Policy](#). Wikipedia® is a registered trademark of the [Wikimedia Foundation, Inc.](#), a non-profit organization.

[Privacy policy](#) [About Wikipedia](#) [Disclaimers](#) [Contact Wikipedia](#) [Developers](#) [Mobile view](#)

