



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  [Add links](#)

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

Article [Talk](#)

[Read](#) [Edit](#) [More▼](#)



# Difference-map algorithm

From Wikipedia, the free encyclopedia  
(Redirected from [Difference map algorithm](#))

The **difference-map algorithm** is a [search algorithm](#) for general [constraint satisfaction](#) problems. It is a [meta-algorithm](#) in the sense that it is built from more basic algorithms that perform [projections](#) onto [constraint](#) sets. From a mathematical perspective, the difference-map algorithm is a [dynamical system](#) based on a [mapping](#) of [Euclidean space](#). Solutions are encoded as [fixed points](#) of the mapping.

Although originally conceived as a general method for solving the [phase problem](#), the difference-map algorithm has been used for the [boolean satisfiability problem](#), [protein structure prediction](#), [Ramsey numbers](#), [diophantine equations](#), and [Sudoku](#),<sup>[1]</sup> as well as sphere- and disk-packing problems.<sup>[2]</sup> Since these applications include [NP-complete](#) problems, the scope of the difference map is that of an [incomplete algorithm](#). Whereas incomplete algorithms can efficiently verify solutions (once a candidate is found), they cannot prove that a solution does not exist.

The difference-map algorithm is a generalization of two [iterative methods](#): Fienup's [Hybrid input output \(HIO\) algorithm for phase retrieval](#)<sup>[3]</sup> and the [Douglas-Rachford algorithm](#)<sup>[4]</sup> for [convex optimization](#). Iterative methods, in general, have a long history in phase retrieval and convex optimization. The use of this style of algorithm for hard, non-convex problems is a more recent development.

## Contents [\[hide\]](#)

- [1 Algorithm](#)
- [2 Example: logical satisfiability](#)
- [3 Chaotic dynamics](#)
- [4 Phase retrieval](#)
- [5 Notes](#)

## Algorithm [\[edit\]](#)

The problem to be solved must first be formulated as a [set intersection](#) problem in Euclidean space: find an *x* in the intersection of sets *A* and *B*. Another prerequisite is an implementation of the projections *P<sub>A</sub>* and *P<sub>B</sub>* that, given an arbitrary input point *x*, return a point in the constraint set *A* or *B* that is nearest to *x*. One iteration of the algorithm is given by the mapping:

$$\begin{aligned}x &\mapsto D(x) = x + \beta \left[ P_A \left( f_B(x) \right) - P_B \left( f_A(x) \right) \right], \\ f_A(x) &= P_A(x) - \frac{1}{\beta} \left( P_A(x) - x \right), \\ f_B(x) &= P_B(x) + \frac{1}{\beta} \left( P_B(x) - x \right)\end{aligned}$$

The real parameter *β* should not be equal to 0 but can have either sign; optimal values depend on the application and are determined through experimentation. As a first guess, the choice *β* = 1 (or *β* = −1) is recommended because it reduces the number of projection computations per iteration:

$$D(x) = x + P_A(2P_B(x) - x) - P_B(x)$$

A point *x* is a fixed point of the map *x*  $\mapsto$  *D*(*x*) precisely when *P<sub>A</sub>*(*f<sub>B</sub>*(*x*)) = *P<sub>B</sub>*(*f<sub>A</sub>*(*x*)). Since the left-hand side is an element of *A* and the RHS is an element of *B*, the equality implies that we have found a common element to the two constraint sets. Note that the fixed point *x* itself need not belong to either *A* or *B*. The set of fixed points will typically have much higher dimension than the set of solutions.

The progress of the algorithm can be monitored by inspecting the norm of the difference of the two projections:

$$\Delta = |P_A(f_B(x)) - P_B(f_A(x))|$$

When this vanishes, a point common to both constraint sets has been found and the algorithm can be terminated.

## Example: logical satisfiability [\[edit\]](#)

Incomplete algorithms, such as stochastic [local search](#), are widely used for finding satisfying truth assignments to boolean formulas. As an example of solving an instance of [2-SAT](#) with the difference-map algorithm, consider the following formula ( $\sim$  indicates NOT):

$$(q_1 \text{ or } q_2) \text{ and } (\sim q_1 \text{ or } q_3) \text{ and } (\sim q_2 \text{ or } \sim q_3) \text{ and } (q_1 \text{ or } \sim q_2)$$

To each of the eight [literals](#) in this formula we assign one real variable in an eight-dimensional Euclidean space. The structure of the 2-SAT formula can be recovered when these variables are arranged in a table:

$x_{11}$	$x_{12}$	
$(x_{21})$		$x_{22}$
	$(x_{31})$	$(x_{32})$
$x_{41}$	$(x_{42})$	

Rows are the clauses in the 2-SAT formula and literals corresponding to the same boolean variable are arranged in columns, with negation indicated by parentheses. For example, the real variables  $x_{11}$ ,  $x_{21}$  and  $x_{41}$  correspond to the same boolean variable ( $q_1$ ) or its negation, and are called **replicas**. It is convenient to associate the values 1 and -1 with *TRUE* and *FALSE* rather than the traditional 1 and 0. With this convention, the compatibility between the replicas takes the form of the following linear equations:

$$x_{11} = -x_{21} = x_{41}$$

$$x_{12} = -x_{31} = -x_{42}$$

$$x_{22} = -x_{32}$$

The linear subspace where these equations are satisfied is one of the constraint spaces, say  $A$ , used by the difference map. To project to this constraint we replace each replica by the signed replica average, or its negative:

$$a_1 = (x_{11} - x_{21} + x_{41}) / 3$$

$$x_{11} \rightarrow a_1 \quad x_{21} \rightarrow -a_1 \quad x_{41} \rightarrow a_1$$

The second difference-map constraint applies to the rows of the table, the clauses. In a satisfying assignment, the two variables in each row must be assigned the values (1, 1), (1, -1), or (-1, 1). The corresponding constraint set,  $B$ , is thus a set of  $3^4 = 81$  points. In projecting to this constraint the following operation is applied to each row. First, the two real values are rounded to 1 or -1; then, if the outcome is (-1, -1), the larger of the two original values is replaced by 1. Examples:

$$(-.2, 1.2) \rightarrow (-1, 1)$$

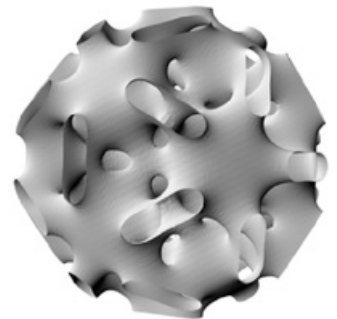
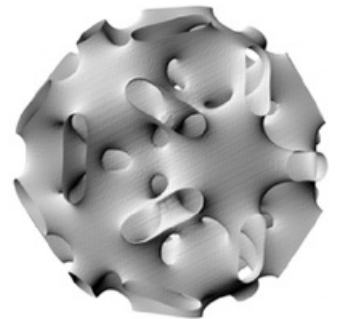
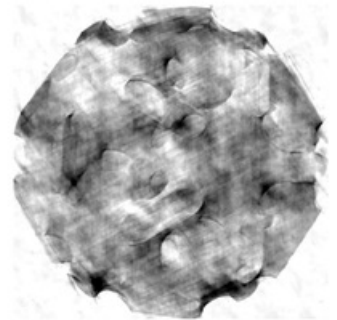
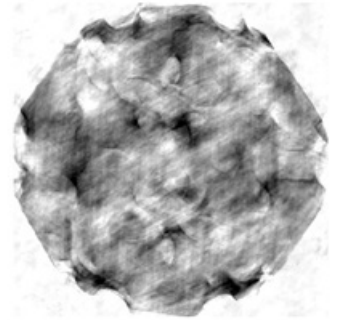
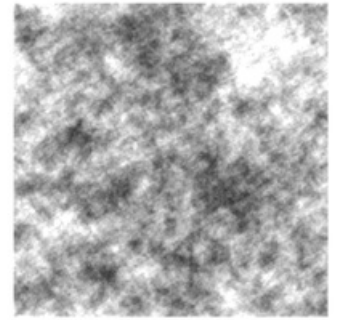
$$(-.2, -.8) \rightarrow (1, -1)$$


It is a straightforward exercise to check that both of the projection operations described minimize the Euclidean distance between input and output values. Moreover, if the algorithm succeeds in finding a point  $x$  that lies in both constraint sets, then we know that (i) the clauses associated with  $x$  are all *TRUE*, and (ii) the assignments to the replicas are consistent with a truth assignment to the original boolean variables.

To run the algorithm one first generates an initial point  $x_0$ , say

-0.5	-0.8	
(-0.4)		-0.6
	(0.3)	(-0.8)
0.5	(0.1)	

Using  $\beta = 1$ , the next step is to compute  $P_B(x_0)$  :



Iterations 0, 100, 200, 300 and 400   
in the difference-map reconstruction of  
a grayscale image from its Fourier  
transform modulus

1	-1	
(1)		-1
	(1)	(-1)
1	(1)	

This is followed by  $2P_B(x_0) - x_0$ ,

2.5	-1.2	
(2.4)		-1.4
	(1.7)	(-1.2)
1.5	(1.9)	

and then projected onto the other constraint,  $P_A(2P_B(x_0) - x_0)$  :

0.53333	-1.6	
(-0.53333)		-0.1
	(1.6)	(0.1)
0.53333	(1.6)	

Incrementing  $x_0$  by the difference of the two projections gives the first iteration of the difference map,  $D(x_0) = x_1$  :

-0.96666	-1.4	
(-1.93333)		0.3
	(0.9)	(0.3)
0.03333	(0.7)	

Here is the second iteration,  $D(x_1) = x_2$  :

-0.3	-1.4	
(-2.6)		-0.7
	(0.9)	(-0.7)
0.7	(0.7)	

This is a fixed point:  $D(x_2) = x_2$ . The iterate is unchanged because the two projections agree. From  $P_B(x_2)$ ,

1	-1	
(-1)		1
	(1)	(-1)
1	(1)	

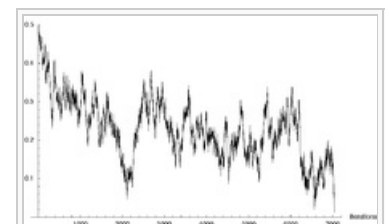
we can read off the satisfying truth assignment:  $q_1 = \text{TRUE}$ ,  $q_2 = \text{FALSE}$ ,  $q_3 = \text{TRUE}$ .

## Chaotic dynamics [\[edit\]](#)

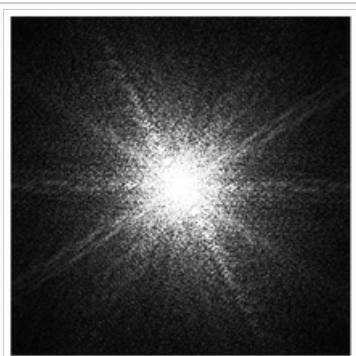
In the simple 2-SAT example above, the norm of the difference-map increment  $\Delta$  decreased monotonically to zero in three iterations. This contrasts the behavior of  $\Delta$  when the difference map is given a hard instance of 3-SAT, where it fluctuates strongly prior to the discovery of the fixed point. As a dynamical system the difference map is believed to be [chaotic](#), and that the space being searched is a [strange attractor](#).

## Phase retrieval [\[edit\]](#)

In phase retrieval a signal or image is reconstructed from the [modulus](#) (absolute value, magnitude) of its [discrete Fourier transform](#). For example, the source of the modulus data may be the [Fraunhofer diffraction](#) pattern formed when an object is illuminated with [coherent light](#).



Time series of the norm of the difference-map increment  $\Delta$  in the course of solving a random 3-SAT instance with 1000 variables and 4200 clauses.



Fourier transform modulus (diffraction pattern) of the grayscale image shown being reconstructed at the top of the page.

The projection to the Fourier modulus constraint, say  $P_A$ , is accomplished by first computing the discrete Fourier transform of the signal or image, rescaling the moduli to agree with the data, and then inverse transforming the result. This is a projection, in the sense that the Euclidean distance to the constraint is minimized, because (i) the discrete Fourier transform, as a [unitary transformation](#), preserves distance, and (ii) rescaling the modulus (without modifying the phase) is the smallest change that realizes the modulus constraint.

To recover the unknown phases of the Fourier transform the difference map relies on the projection to another constraint,  $P_B$ . This may take several forms, as the object being reconstructed may be known to be positive, have a bounded [support](#), etc. In the reconstruction of the surface image, for example, the effect of the projection  $P_B$  was to nullify all values outside a rectangular support, and also to nullify all negative values within the support.

## Notes [\[edit\]](#)

- <sup>^</sup> V. Elser, I. Rankenburg, and P. Thibault, "Searching with iterated maps". *Proceedings of the National Academy of Sciences USA*. (2007). **104**:418-423. <http://www.pnas.org/cgi/content/short/104/2/418>
- <sup>^</sup> S. Gravel, V. Elser, "Divide and conquer: A general approach to constraint satisfaction". *Physical Review E*. (2008). **78**:036706. <http://link.aps.org/doi/10.1103/PhysRevE.78.036706>
- <sup>^</sup> J.R. Fienup, "Phase retrieval algorithms: a comparison". *Applied Optics*. (1982). **21**:2758-2769.
- <sup>^</sup> H.H. Bauschke, P.L. Combettes, and D.R. Luke, "Phase retrieval, error reduction algorithm, and Fienup variants: a view from convex optimization". *Journal of the Optical Society of America A*. (2002). **19**:1334-1345.

Categories: [Search algorithms](#) | [Constraint programming](#)

This page was last modified on 29 July 2014, at 07:40.

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](#)

