



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

[Deutsch](#)

[Français](#)

[Русский](#)

[Edit links](#)

Article [Talk](#)

[Read](#)

[Edit](#)

[View history](#)

Search

# Biconjugate gradient method

From Wikipedia, the free encyclopedia



This article includes a [list of references](#), related reading or [external links](#), but its sources remain unclear because it lacks [inline citations](#).

Please [improve](#) this article by introducing more precise citations.

(September 2013)

In [mathematics](#), more specifically in [numerical linear algebra](#), the **biconjugate gradient method** is an [algorithm](#) to solve [systems of linear equations](#)

$$Ax = b.$$

Unlike the [conjugate gradient method](#), this algorithm does not require the [matrix](#) *A* to be [self-adjoint](#), but instead one needs to perform multiplications by the [conjugate transpose](#) *A*<sup>\*</sup>.

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

- The algorithm
  - Unpreconditioned version of the algorithm
- Discussion
- Properties
- See also
- References

## The algorithm [\[edit\]](#)

- Choose initial guess *x*<sub>0</sub>, two other vectors *x*<sub>0</sub><sup>\*</sup> and *b*<sup>\*</sup> and a [preconditioner](#) *M*
- r*<sub>0</sub> ← *b* − *A* *x*<sub>0</sub>
- r*<sub>0</sub><sup>\*</sup> ← *b*<sup>\*</sup> − *x*<sub>0</sub><sup>\*</sup> *A*<sup>T</sup>
- p*<sub>0</sub> ← *M*<sup>−1</sup> *r*<sub>0</sub>
- p*<sub>0</sub><sup>\*</sup> ← *r*<sub>0</sub><sup>\*</sup> *M*<sup>−1</sup>
- for *k* = 0, 1, . . . do
  - α*<sub>*k*</sub> ←  $\frac{r_k^* M^{-1} r_k}{p_k^* A p_k}$
  - x*<sub>*k*+1</sub> ← *x*<sub>*k*</sub> + *α*<sub>*k*</sub> · *p*<sub>*k*</sub>
  - x*<sub>*k*+1</sub><sup>\*</sup> ← *x*<sub>*k*</sub><sup>\*</sup> +  $\overline{\alpha_k}$  · *p*<sub>*k*</sub><sup>\*</sup>
  - r*<sub>*k*+1</sub> ← *r*<sub>*k*</sub> − *α*<sub>*k*</sub> · *A* *p*<sub>*k*</sub>
  - r*<sub>*k*+1</sub><sup>\*</sup> ← *r*<sub>*k*</sub><sup>\*</sup> −  $\overline{\alpha_k}$  · *p*<sub>*k*</sub><sup>\*</sup> *A*
  - β*<sub>*k*</sub> ←  $\frac{r_{k+1}^* M^{-1} r_{k+1}}{r_k^* M^{-1} r_k}$
  - p*<sub>*k*+1</sub> ← *M*<sup>−1</sup> *r*<sub>*k*+1</sub> + *β*<sub>*k*</sub> · *p*<sub>*k*</sub>
  - p*<sub>*k*+1</sub><sup>\*</sup> ← *r*<sub>*k*+1</sub><sup>\*</sup> *M*<sup>−1</sup> +  $\overline{\beta_k}$  · *p*<sub>*k*</sub><sup>\*</sup>

In the above formulation, the computed *r*<sub>*k*</sub> and *r*<sub>*k*</sub><sup>\*</sup> satisfy

$$\begin{aligned} r_k &= b - Ax_k, \\ r_k^* &= b^* - x_k^* A \end{aligned}$$

and thus are the respective [residuals](#) corresponding to *x*<sub>*k*</sub> and *x*<sub>*k*</sub><sup>\*</sup>, as approximate solutions to the systems

$$\begin{aligned} Ax &= b, \\ x^* A &= b^*; \end{aligned}$$

*x*<sup>\*</sup> is the [adjoint](#), and  $\overline{\alpha}$  is the [complex conjugate](#).

### Unpreconditioned version of the algorithm [\[edit\]](#)

1. Choose initial guess  $x_0$ ,
2.  $r_0 \leftarrow b - Ax_0$
3.  $\hat{r}_0 \leftarrow \hat{b} - \hat{x}_0 A^T$
4.  $p_0 \leftarrow r_0$
5.  $\hat{p}_0 \leftarrow \hat{r}_0$
6. for  $k = 0, 1, \dots$  do
  1.  $\alpha_k \leftarrow \frac{\hat{r}_k r_k}{\hat{p}_k A p_k}$
  2.  $x_{k+1} \leftarrow x_k + \alpha_k \cdot p_k$
  3.  $\hat{x}_{k+1} \leftarrow \hat{x}_k + \alpha_k \cdot \hat{p}_k$
  4.  $r_{k+1} \leftarrow r_k - \alpha_k \cdot A p_k$
  5.  $\hat{r}_{k+1} \leftarrow \hat{r}_k - \alpha_k \cdot \hat{p}_k A^T$
  6.  $\beta_k \leftarrow \frac{\hat{r}_{k+1} r_{k+1}}{\hat{r}_k r_k}$
  7.  $p_{k+1} \leftarrow r_{k+1} + \beta_k \cdot p_k$
  8.  $\hat{p}_{k+1} \leftarrow \hat{r}_{k+1} + \beta_k \cdot \hat{p}_k$

## Discussion [\[edit\]](#)

The biconjugate gradient method is [numerically unstable](#)<sup>[\[citation needed\]](#)</sup> (compare to the [biconjugate gradient stabilized method](#)), but very important from a theoretical point of view. Define the iteration steps by

$$\begin{aligned} x_k &:= x_j + P_k A^{-1} (b - Ax_j), \\ x_k^* &:= x_j^* + (b^* - x_j^* A) P_k A^{-1}, \end{aligned}$$

where  $j < k$  using the related [projection](#)

$$P_k := \mathbf{u}_k (\mathbf{v}_k^* A \mathbf{u}_k)^{-1} \mathbf{v}_k^* A,$$

with

$$\begin{aligned} \mathbf{u}_k &= [u_0, u_1, \dots, u_{k-1}], \\ \mathbf{v}_k &= [v_0, v_1, \dots, v_{k-1}]. \end{aligned}$$

These related projections may be iterated themselves as

$$P_{k+1} = P_k + (1 - P_k) u_k \otimes \frac{v_k^* A (1 - P_k)}{v_k^* A (1 - P_k) u_k}.$$

A relation to [Quasi-Newton methods](#) is given by  $P_k = A_k^{-1} A$  and  $x_{k+1} = x_k - A_{k+1}^{-1} (Ax_k - b)$ , where

$$A_{k+1}^{-1} = A_k^{-1} + (1 - A_k^{-1} A) u_k \otimes \frac{v_k^* (1 - A A_k^{-1})}{v_k^* A (1 - A_k^{-1} A) u_k}.$$

The new directions

$$\begin{aligned} p_k &= (1 - P_k) u_k, \\ p_k^* &= v_k^* A (1 - P_k) A^{-1} \end{aligned}$$

are then orthogonal to the residuals:

$$\begin{aligned} v_i^* r_k &= p_i^* r_k = 0, \\ r_k^* u_j &= r_k^* p_j = 0, \end{aligned}$$

which themselves satisfy

$$\begin{aligned} r_k &= A (1 - P_k) A^{-1} r_j, \\ r_k^* &= r_j^* (1 - P_k) \end{aligned}$$

where  $i, j < k$ .

The biconjugate gradient method now makes a special choice and uses the setting

$$\begin{aligned} u_k &= M^{-1} r_k, \\ v_k^* &= r_k^* M^{-1}. \end{aligned}$$

With this particular choice, explicit evaluations of  $P_k$  and  $A^{-1}$  are avoided, and the algorithm takes the form stated above.

## Properties [\[edit\]](#)

- If  $A = A^*$  is **self-adjoint**,  $x_0^* = x_0$  and  $b^* = b$ , then  $r_k = r_k^*$ ,  $p_k = p_k^*$ , and the **conjugate gradient method** produces the same sequence  $x_k = x_k^*$  at half the computational cost.
- The sequences produced by the algorithm are **biorthogonal**, i.e.,  $p_i^* A p_j = r_i^* M^{-1} r_j = 0$  for  $i \neq j$ .
- If  $P_{j'}$  is a polynomial with  $\deg(P_{j'}) + j < k$ , then  $r_k^* P_{j'} (M^{-1} A) u_j = 0$ . The algorithm thus produces projections onto the **Krylov subspace**.
- If  $P_{i'}$  is a polynomial with  $i + \deg(P_{i'}) < k$ , then  $v_i^* P_{i'} (A M^{-1}) r_k = 0$ .

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

- Biconjugate gradient stabilized method
- Conjugate gradient method

## References [\[edit\]](#)

- Fletcher, R. (1976). Watson, G. Alistair, ed. "[Conjugate gradient methods for indefinite systems](#)". *Numerical Analysis*. Lecture Notes in Mathematics (Springer Berlin / Heidelberg) **506**: 73–89. doi:[10.1007/BFb0080109](#). ISBN 978-3-540-07610-0. ISSN 1617-9692.
- Press, WH; Teukolsky, SA; Vetterling, WT; Flannery, BP (2007). "[Section 2.7.6](#)". *Numerical Recipes: The Art of Scientific Computing* (3rd ed.). New York: Cambridge University Press. ISBN 978-0-521-88068-8

v · t · e		Numerical linear algebra	[hide]
Key concepts	Floating point · Numerical stability		
Problems	Matrix multiplication (algorithms) · Matrix decompositions · Linear equations · Sparse problems		
Hardware	CPU cache · TLB · Cache-oblivious algorithm · SIMD · Multiprocessing		
Software	BLAS · Specialized libraries · General purpose software		

Categories: Numerical linear algebra | Gradient methods

This page was last modified on 13 November 2013, at 20:39.

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](#)

