Gaussian elimination

From 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
Alemannisch
العربية
Azərbaycanca
Català
Čeština
★ Deutsch
Esperanto
فارسی
Français
한국어
Հայերեն
हिन्दी
Íslenska
Italiano
עברית
Latviešu
Magyar
Nederlands
日本語
Norsk bokmål
Polski
Português
Русский
తెలుగు
Simple English
Slovenščina
Srpskohrvatski / српскохрватски
Suomi
Svenska
Українська
Vepsän kel'
Tiếng Việt
中文
Edit links

In linear algebra, **Gaussian elimination** (also known as **row reduction**) is an algorithm for solving systems of linear equations. It is usually understood as a sequence of operations performed on the associated matrix of coefficients. This method can also be used to find the rank of a matrix, to calculate the determinant of a matrix, and to calculate the inverse of an invertible square matrix. The method is named after Carl Friedrich Gauss (1777–1855), although it was known to Chinese mathematicians as early as 179 CE (see History section).

To perform row reduction on a matrix, one uses a sequence of elementary row operations to modify the matrix until the lower left-hand corner of the matrix is filled with zeros, as much as possible. There are three types of elementary row operations: 1) Swapping two rows, 2) Multiplying a row by a non-zero number, 3) Adding a multiple of one row to another row. Using these operations, a matrix can always be transformed into an upper triangular matrix, and in fact one that is in row echelon form. Once all of the leading coefficients (the left-most non-zero entry in each row) are 1, and in every column containing a leading coefficient has zeros elsewhere, the matrix is said to be in reduced row echelon form. This final form is unique; in other words, it is independent of the sequence of row operations used. For example, in the following sequence of row operations (where multiple elementary operations might be done at each step), the third and fourth matrices are the ones in row echelon form, and the final matrix is the unique reduced row echelon form.

$$\left[\begin{array}{ccc|c} 1 & 3 & 1 & 9 \\ 1 & 1 & -1 & 1 \\ 3 & 11 & 5 & 35 \end{array}\right] \rightarrow \left[\begin{array}{ccc|c} 1 & 3 & 1 & 9 \\ 0 & -2 & -2 & -8 \\ 0 & 2 & 2 & 8 \end{array}\right] \rightarrow \left[\begin{array}{ccc|c} 1 & 3 & 1 & 9 \\ 0 & -2 & -2 & -8 \\ 0 & 0 & 0 & 0 \end{array}\right] \rightarrow \left[\begin{array}{ccc|c} 1 & 0 & -2 & -3 \\ 0 & 1 & 1 & 4 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

Using row operations to convert a matrix into reduced row echelon form is sometimes called **Gauss–Jordan elimination**. Some authors use the term Gaussian elimination to refer to the process until it has reached its upper triangular, or (non-reduced) row echelon form. For computational reasons, when solving systems of linear equations, it is sometimes preferable to stop row operations before the matrix is completely reduced.

Contents [hide]

Definitions and example of algorithm [edit]

The process of row reduction makes use of elementary row operations, and can be divided into two parts. The first part (sometimes called Forward Elimination) reduces a given system to *row echelon form*, from which one can tell whether there are no solutions, a unique solution, or infinitely many solutions. The second part (sometimes called back substitution) continues to use row operations until the solution is found; in other words, it puts the matrix into *reduced* row echelon form.

Another point of view, which turns out to be very useful to analyze the algorithm, is that row reduction produces a matrix decomposition of the original matrix. The elementary row operations may be viewed as the multiplication on the left of the original matrix by elementary matrices. Alternatively, a sequence of elementary operations that reduces a single row may be viewed as multiplication by a Frobenius matrix. Then the first part of the algorithm computes an LU decomposition, while the second part writes the original matrix as the product of a uniquely determined invertible matrix and a uniquely determined reduced row echelon matrix.

Row operations [edit]

See also: Elementary matrix

There are three types of **elementary row operations** which may be performed on the rows of a matrix:

Type 1: Swap the positions of two rows.

Type 2: Multiply a row by a nonzero scalar.

Type 3: Add to one row a scalar multiple of another.

If the matrix is associated to a system of linear equations, then these operations do not change the solution set. Therefore, if one's goal is to solve a system of linear equations, then using these row operations could make the problem easier.

Echelon form [edit]

Main article: Row echelon form

For each row in a matrix, if the row does not consist of only zeros, then the left-most non-zero entry is called the *leading coefficient* (or *pivot*) of that row. So if two leading coefficients are in the same column, then a row operation of type 3 (see above) could be used to make one of those coefficients zero. Then by using the row swapping operation, one can always order the rows so that for every non-zero row, the leading coefficient is to the right of the leading coefficient of the row above. If this is the case, then matrix is said to be in **row echelon form**. So the lower left part of the matrix contains only zeros, and all of the zero rows are below the non-zero rows. The word "echelon" is used here because one can roughly think of the rows being ranked by their size, with the largest being at the top and the smallest being at the bottom.

For example, the following matrix is in row echelon form, and its leading coefficients are shown in red.

$$\begin{bmatrix} 0 & \mathbf{2} & 1 & -1 \\ 0 & 0 & \mathbf{3} & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

It is in echelon form because the zero row is at the bottom, and the leading coefficient of the second row (in the third column), is to the right of the leading coefficient of the first row (in the second column).

A matrix is said to be in **reduced row echelon form** if furthermore all of the leading coefficients are equal to 1 (which can be achieved by using the elementary row operation of type 2), and in every column containing a leading coefficient, all of the other entries in that column are zero (which can be achieved by using elementary row operations of type 3).

Example of the algorithm [edit]

Suppose the goal is to find and describe the set of solutions to the following system of linear equations:

$$\begin{aligned} 2x + y - z &= 8 && (L_1) \\ -3x - y + 2z &= -11 && (L_2) \\ -2x + y + 2z &= -3 && (L_3) \end{aligned}$$

The table below is the row reduction process applied simultaneously to the system of equations, and its associated augmented matrix. In practice, one does not usually deal with the systems in terms of equations but instead makes use of the augmented matrix, which is more suitable for computer manipulations. The row reduction procedure may be summarized as follows: eliminate *x* from all equations below L_1, and then eliminate *y* from all equations below L_2. This will put the system into triangular form. Then, using back-substitution, each unknown can be solved for.

System of equations	Row operations	Augmented matrix
$\begin{aligned} 2x + y - z &= 8 \\ -3x - y + 2z &= -11 \\ -2x + y + 2z &= -3 \end{aligned}$		$\left[\begin{array}{ccc\|c} 2 & 1 & -1 & 8 \\ -3 & -1 & 2 & -11 \\ -2 & 1 & 2 & -3 \end{array}\right]$
$\begin{aligned} 2x + y - z &= 8 \\ \frac{1}{2}y + \frac{1}{2}z &= 1 \\ 2y + z &= 5 \end{aligned}$	$\begin{aligned} L_2 + \frac{3}{2}L_1 &\to L_2 \\ L_3 + L_1 &\to L_3 \end{aligned}$	$\left[\begin{array}{ccc\|c} 2 & 1 & -1 & 8 \\ 0 & 1/2 & 1/2 & 1 \\ 0 & 2 & 1 & 5 \end{array}\right]$
$\begin{aligned} 2x + y - z &= 8 \\ \frac{1}{2}y + \frac{1}{2}z &= 1 \\ -z &= 1 \end{aligned}$	$L_3 + -4L_2 \to L_3$	$\left[\begin{array}{ccc\|c} 2 & 1 & -1 & 8 \\ 0 & 1/2 & 1/2 & 1 \\ 0 & 0 & -1 & 1 \end{array}\right]$
The matrix is now in echelon form (also called triangular form)		
	$\begin{aligned} L_2 + \frac{1}{2}L_3 &\to L_2 \\ L_1 - L_3 &\to L_1 \end{aligned}$	$\left[\begin{array}{ccc\|c} 2 & 1 & 0 & 7 \\ 0 & 1/2 & 0 & 3/2 \\ 0 & 0 & -1 & 1 \end{array}\right]$

$\begin{aligned} 2x + y &= 7 \\ \tfrac{1}{2}y &= 3/2 \\ z &= -1 \end{aligned}$	$\begin{aligned} 2L_2 &\to L_2 \\ -L_3 &\to L_3 \end{aligned}$	$\left[\begin{array}{ccc\|c} 2 & 1 & 0 & 7 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & -1 \end{array}\right]$
$\begin{aligned} x &= 2 \\ y &= 3 \\ z &= -1 \end{aligned}$	$\begin{aligned} L_1 - L_2 &\to L_1 \\ \tfrac{1}{2}L_1 &\to L_1 \end{aligned}$	$\left[\begin{array}{ccc\|c} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & -1 \end{array}\right]$

The second column describes which row operations have just been performed. So for the first step, the *x* is eliminated from L_2 by adding $\frac{3}{2}L_1$ to L_2. Next *x* is eliminated from L_3 by adding L_1 to L_3. These row operations are labelled in the table as

$$L_2 + \frac{3}{2}L_1 \to L_2$$
$$L_3 + L_1 \to L_3.$$

Once *y* is also eliminated from the third row, the result is a system of linear equations in triangular form, and so the first part of the algorithm is complete. From a computational point of view, it is faster to solve the variables in reverse order, a process known as back-substitution. One sees the solution is z = -1, y = 3, and x = 2. So there is a unique solution to the original system of equations.

Instead of stopping once the matrix is in echelon form, one could continue until the matrix is in *reduced* row echelon form, as it is done in the table. The process of row reducing until the matrix is reduced is sometimes referred to as **Gauss-Jordan elimination**, to distinguish it from stopping after reaching echelon form.

History [edit]

The method of Gaussian elimination appears in the Chinese mathematical text Chapter Eight *Rectangular Arrays* of *The Nine Chapters on the Mathematical Art*. Its use is illustrated in eighteen problems, with two to five equations. The first reference to the book by this title is dated to 179 CE, but parts of it were written as early as approximately 150 BCE.[1][2] It was commented on by Liu Hui in the 3rd century.

The method in Europe stems from the notes of Isaac Newton.[3][4] In 1670, he wrote that all the algebra books known to him lacked a lesson for solving simultaneous equations, which Newton then supplied. Cambridge University eventually published the notes as *Arithmetica Universalis* in 1707 long after Newton left academic life. The notes were widely imitated, which made (what is now called) Gaussian elimination a standard lesson in algebra textbooks by the end of the 18th century. Carl Friedrich Gauss in 1810 devised a notation[*which?*] for symmetric elimination that was adopted in the 19th century by professional hand computers to solve the normal equations of least-squares problems. The algorithm that is taught in high school was named for Gauss only in the 1950s as a result of confusion over the history of the subject.[5]

Some authors use the term *Gaussian elimination* to refer only to the procedure until the matrix is in echelon form, and use the term **Gauss-Jordan elimination** to refer to the procedure which ends in reduced echelon form. The name is used because it is a variation of Gaussian elimination as described by Wilhelm Jordan in 1887. However, the method also appears in an article by Clasen published in the same year. Jordan and Clasen probably discovered Gauss–Jordan elimination independently.[6]

Applications [edit]

The historically first application of the row reduction method is for solving systems of linear equations. Here are some other important applications of the algorithm.

Computing determinants [edit]

To explain how Gaussian elimination allows to compute the determinant of a square matrix, we have to recall how the elementary row operations change the determinant:

- Swapping two rows multiplies the determinant by -1
- Multiplying a row by a nonzero scalar multiplies the determinant by the same scalar
- Adding to one row a scalar multiple of another does not change the determinant.

If the Gaussian elimination applied to a square matrix *A* produces a row echelon matrix *B*, let *d* be the product of the scalars by which the determinant has been multiplied, using above rules. Then the determinant of *A* is the quotient by *d* of the product of the elements of the diagonal of *B*: det(*A*) = ∏diag(*B*) / *d*.

Computationally, for a *n*×*n* matrix, this method needs only $O(n^3)$ arithmetic operations, while solving by elementary methods requires $O(2^n)$ or $O(n!)$ operations. Even on the fastest computers, the elementary methods are impractical for *n* above 20.

Finding the inverse of a matrix [edit]

See also: Invertible matrix

A variant of Gaussian elimination called Gauss–Jordan elimination can be used for finding the inverse of a matrix, if it exists. If *A* is a *n* by *n* square matrix, then one can use row reduction to compute its inverse matrix, if it exists. First, the *n* by *n* identity matrix is augmented to the right of *A*, forming a *n* by 2*n* block matrix [*A* | *I*]. Now through application of elementary row operations, find the reduced echelon form of this *n* by *2n* matrix. The matrix *A* is invertible if and only if the left block can be reduced to the identity matrix *I*; in this case the right block of the final matrix is A^{-1}. If the algorithm is unable to reduce the left block to *I*, then *A* is not invertible.

For example, consider the following matrix

$$A = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}.$$

To find the inverse of this matrix, one takes the following matrix augmented by the identity, and row reduces it as a 3 by 6 matrix:

$$[A|I] = \left[\begin{array}{ccc|ccc} 2 & -1 & 0 & 1 & 0 & 0 \\ -1 & 2 & -1 & 0 & 1 & 0 \\ 0 & -1 & 2 & 0 & 0 & 1 \end{array}\right].$$

By performing row operations, one can check that the reduced row echelon form of this augmented matrix is:

$$[I|B] = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & \frac{3}{4} & \frac{1}{2} & \frac{1}{4} \\ 0 & 1 & 0 & \frac{1}{2} & 1 & \frac{1}{2} \\ 0 & 0 & 1 & \frac{1}{4} & \frac{1}{2} & \frac{3}{4} \end{array}\right].$$

One can think of each row operation as an elementary matrix, the product of which can be called *B*. On the left, we showed that *AB* = *I*, and therefore, *B* = A^{-1}. On the right, we kept a record of *IB* = *B*, which we know is the inverse desired. This procedure for finding the inverse works for square matrices of any size.

Computing ranks and bases [edit]

The Gaussian elimination algorithm can be applied to any $m \times n$ matrix A. In this way, for example, some 6×9 matrices can be transformed to a matrix that has a row echelon form like

$$T = \begin{bmatrix} a & * & * & * & * & * & * & * & * \\ 0 & 0 & b & * & * & * & * & * & * \\ 0 & 0 & 0 & c & * & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 & d & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & e \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

where the *s are arbitrary entries and *a, b, c, d, e* are nonzero entries. This echelon matrix T contains a wealth of information about A: the rank of A is 5 since there are 5 non-zero rows in T; the vector space spanned by the columns of A has a basis consisting of the first, third, fourth, seventh and ninth column of A (the columns of *a, b, c, d, e* in T), and the *s tell you how the other columns of A can be written as linear combinations of the basis columns. This is a consequence of the distributivity of the dot product in the expression of a linear map as a matrix.

All of this applies also to the reduced row echelon form, which is a particular row echelon form.

Computational efficiency [edit]

The number of arithmetic operations required to perform row reduction is one way of measuring the algorithm's computational efficiency. For example, to solve a system of *n* equations for *n* unknowns by performing row operations on the matrix until it is in echelon form, and then solving for each unknown in reverse order, requires $n(n+1) / 2$ divisions, $(2n^3 + 3n^2 - 5n)/6$ multiplications, and $(2n^3 + 3n^2 - 5n)/6$ subtractions,[7] for a total of approximately $2n^3 / 3$ operations. Thus it has arithmetic complexity of $O(n^3)$; see Big O notation. This arithmetic complexity is a good measure of the time needed for the whole computation when the time for each arithmetic operation is approximatively constant. This is the case when the coefficients are represented by floating point numbers or when they belong to a finite field. If the coefficients are integers or rational numbers exactly represented, the intermediate entries can grow exponentially large, so the bit complexity is exponential.[8] However, there is a variant of Gaussian elimination, called Bareiss algorithm that avoids this exponential growth of the intermediate entries, and, with the same arithmetic complexity of $O(n^3)$, has a bit complexity of $O(n^5)$.

This algorithm can be used on a computer for systems with thousands of equations and unknowns. However, the cost becomes prohibitive for systems with millions of equations. These large systems are generally solved using iterative methods. Specific methods exist for systems whose coefficients follow a regular pattern (see system of linear equations).

To put an *n* by *n* matrix into reduced echelon form by row operations, one needs n^3 arithmetic operations; which is approximately 50% more computation steps.[9]

One possible problem is numerical instability, caused by the possibility of dividing by very small numbers. If, for example, the leading coefficient of one of the rows is very close to zero, then to row reduce the matrix one would need to divide by

that number so the leading coefficient is 1. This means any error that existed for the number which was close to zero would be amplified. Gaussian elimination is numerically stable for diagonally dominant or positive-definite matrices. For general matrices, Gaussian elimination is usually considered to be stable, when using partial pivoting, even though there are examples of stable matrices for which it is unstable.[10]

Generalizations [edit]

The Gaussian elimination can be performed over any field, not just the real numbers.

Gaussian elimination does not generalize in any simple way to higher order tensors (matrices are array representations of order 2 tensors); even computing the rank of a tensor of order greater than 2 is a difficult problem.

Pseudocode [edit]

As explained above, Gaussian elimination writes a given $m \times n$ matrix A uniquely as a product of an invertible $m \times m$ matrix S and a row-echelon matrix T. Here, S is the product of the matrices corresponding to the row operations performed.

The formal algorithm to compute T from A follows. We write $A[i, j]$ for the entry in row i, column j in matrix A with 1 being the first index. The transformation is performed *in place*, meaning that the original matrix A is lost and successively replaced by T.

```
for k = 1 ... min(m,n):
  Find the k-th pivot:
  i_max  := argmax (i = k ... m, abs(A[i, k]))
  if A[i_max, k] = 0
    error "Matrix is singular!"
  swap rows(k, i_max)
  Do for all rows below pivot:
  for i = k + 1 ... m:
    Do for all remaining elements in current row:
    for j = k + 1 ... n:
      A[i, j]  := A[i, j] - A[k, j] * (A[i, k] / A[k, k])
    Fill lower triangular matrix with zeros:
    A[i, k]  := 0
```

This algorithm differs slightly from the one discussed earlier, because before eliminating a variable, it first exchanges rows to move the entry with the largest absolute value to the pivot position. Such *partial pivoting* improves the numerical stability of the algorithm; some other variants are used.

Upon completion of this procedure the augmented matrix will be in row-echelon form and may be solved by back-substitution.

With modern computers, Gaussian elimination is not always the fastest algorithm to compute the row echelon form of matrix. There are computer libraries, like BLAS, that exploit the specifics of the computer hardware and of the structure of the matrix to choose the best algorithm automatically.

Notes [edit]

1. ^ Calinger (1999), pp. 234–236
2. ^ Timothy Gowers; June Barrow-Green; Imre Leader (8 September 2008). *The Princeton Companion to Mathematics*. Princeton University Press. p. 607. ISBN 978-0-691-11880-2.
3. ^ Grcar (2011a), pp. 169-172
4. ^ Grcar (2011b), pp. 783-785
5. ^ Grcar (2011b), p. 789
6. ^ Althoen, Steven C.; McLaughlin, Renate (1987), "Gauss–Jordan reduction: a brief history", *The American Mathematical Monthly* (Mathematical Association of America) **94** (2): 130–142, doi:10.2307/2322413, ISSN 0002-9890, JSTOR 2322413
7. ^ Farebrother (1988), p. 12
8. ^ Fang, Xin Gui; Havas, George (1997). "On the worst-case complexity of integer Gaussian elimination" (PDF). *Proceedings of the 1997 international symposium on Symbolic and algebraic computation*. ISSAC '97. Kihei, Maui, Hawaii, United States: ACM. pp. 28–31. doi:10.1145/258726.258740. ISBN 0-89791-875-4.
9. ^ J. B. Fraleigh and R. A. Beauregard, Linear Algebra. Addison-Wesley Publishing Company, 1995, Chapter 10
10. ^ Golub & Van Loan (1996), §3.4.6

References [edit]

- Atkinson, Kendall A. (1989), *An Introduction to Numerical Analysis* (2nd ed.), New York: John Wiley & Sons, ISBN 978-0-471-50023-0.
- Bolch, Gunter; Greiner, Stefan; de Meer, Hermann; Trivedi, Kishor S. (2006), *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications* (2nd ed.), Wiley-Interscience, ISBN 978-0-

471-79156-0.
- Calinger, Ronald (1999), *A Contextual History of Mathematics*, Prentice Hall, ISBN 978-0-02-318285-3.
- Farebrother, R.W. (1988), *Linear Least Squares Computations*, STATISTICS: Textbooks and Monographs, Marcel Dekker, ISBN 978-0-8247-7661-9.
- Golub, Gene H.; Van Loan, Charles F. (1996), *Matrix Computations* (3rd ed.), Johns Hopkins, ISBN 978-0-8018-5414-9.
- Grcar, Joseph F. (2011a), "How ordinary elimination became Gaussian elimination", *Historia Mathematica* **38** (2): 163–218, arXiv:0907.2397, doi:10.1016/j.hm.2010.06.003
- Grcar, Joseph F. (2011b), "Mathematicians of Gaussian elimination" (PDF), *Notices of the American Mathematical Society* **58** (6): 782–792
- Higham, Nicholas (2002), *Accuracy and Stability of Numerical Algorithms* (2nd ed.), SIAM, ISBN 978-0-89871-521-7.
- Katz, Victor J. (2004), *A History of Mathematics, Brief Version*, Addison-Wesley, ISBN 978-0-321-16193-2.
- Kaw, Autar; Kalu, Egwu (2010). "Numerical Methods with Applications" (1st ed.). [1]. , Chapter 5 deals with Gaussian elimination.
- Lipson, Marc; Lipschutz, Seymour (2001), *Schaum's outline of theory and problems of linear algebra*, New York: McGraw-Hill, pp. 69–80, ISBN 978-0-07-136200-9.
- Press, WH; Teukolsky, SA; Vetterling, WT; Flannery, BP (2007), "Section 2.2", *Numerical Recipes: The Art of Scientific Computing* (3rd ed.), New York: Cambridge University Press, ISBN 978-0-521-88068-8

External links [edit]

- An online tool for performing elementary operations with rows and columns
- WebApp descriptively solving systems of linear equations with Gaussian Elimination
- A program that performs Gaussian elimination similarly to a human working on paper Exact solutions to systems with rational coefficients.
- Gaussian elimination www.math-linux.com.
- Gaussian elimination calculator with Linear Algebra tutorial.
- Gaussian elimination at Holistic Numerical Methods Institute
- LinearEquations.c Gaussian elimination implemented using C language
- Gauss–Jordan elimination Step by step solution of 3 equations with 3 unknowns using the All-Integer Echelon Method
- Gauss-Jordan Elimination calculator for *n* by *m* matrices, giving intermediate steps
- WildLinAlg13: Solving a system of linear equations on YouTube provides a very clear, elementary presentation of the method of row reduction.

The Wikibook *Linear Algebra* has a page on the topic of: ***Gaussian elimination***

v · t · e **Linear algebra** [hide]

Basic concepts	Scalar · Vector · Vector space · Vector projection · Linear span · Linear map · Linear projection · Linear independence · Linear combination · Basis · Column space · Row space · Dual space · Orthogonality · Kernel · Eigenvalues and eigenvectors · Least squares regressions · Outer product · Inner product space · Dot product · Transpose · Gram–Schmidt process · Linear equations
Matrices	Block · Decomposition · Invertible · Minor · Multiplication · Rank · Transformation · Cramer's rule · **Gaussian elimination**
Numerical	Floating point · Matrix Laboratory · Numerical stability · Basic Linear Algebra Subprograms (BLAS) · Sparse matrix · Comparison of linear algebra libraries · Comparison of numerical analysis software

Categories: Numerical linear algebra | Exchange algorithms | German inventions