

**polyeval()** — Manipulate and evaluate polynomials

<a href="#">Description</a>	<a href="#">Syntax</a>	<a href="#">Remarks and examples</a>	<a href="#">Conformability</a>
<a href="#">Diagnostics</a>	<a href="#">Also see</a>		

## Description

`polyeval(c, x)` evaluates polynomial  $c$  at each value recorded in  $x$ , returning the results in a  $p$ -conformable-with- $x$  vector. For instance, `polyeval((4,2,1), (3\5))` returns  $(4+2*3+3^2 \setminus 4+2*5+5^2) = (19\39)$ .

`polysolve(y, x)` returns the minimal-degree polynomial  $c$  fitting  $y = \text{polyeval}(c, x)$ . Solution is via Lagrange's interpolation formula.

`polytrim(c)` returns polynomial  $c$  with trailing zeros removed. For instance, `polytrim((1,2,3,0))` returns  $(1,2,3)$ . `polytrim((0,0,0,0))` returns  $(0)$ . Thus if  $n = \text{cols}(\text{polytrim}(c))$ , then  $c$  records an  $(n - 1)$ th degree polynomial.

`polyderiv(c, i)` returns the polynomial that is the  $i$ th derivative of polynomial  $c$ . For instance, `polyderiv((4,2,1), 1)` returns  $(2,2)$  (the derivative of  $4 + 2x + x^2$  is  $2 + 2x$ ). The value of the first derivative of polynomial  $c$  at  $x$  is `polyeval(polyderiv(c,1), x)`.

`polyinteg(c, i)` returns the polynomial that is the  $i$ th integral of polynomial  $c$ . For instance, `polyinteg((4,2,1), 1)` returns  $(0,4,1, .3333)$  (the integral of  $4+2x+x^2$  is  $0+4x+x^2+.3333x^3$ ). The value of the integral of polynomial  $c$  at  $x$  is `polyeval(polyinteg(c,1), x)`.

`polyadd(c1, c2)` returns the polynomial that is the sum of the polynomials  $c_1$  and  $c_2$ . For instance, `polyadd((2,1), (3,5,1))` is  $(5,6,1)$  (the sum of  $2 + x$  and  $3 + 5x + x^2$  is  $5 + 6x + x^2$ ).

`polymult(c1, c2)` returns the polynomial that is the product of the polynomials  $c_1$  and  $c_2$ . For instance, `polymult((2,1), (3,5,1))` is  $(6,13,7,1)$  (the product of  $2 + x$  and  $3 + 5x + x^2$  is  $6 + 13x + 7x^2 + x^3$ ).

`polydiv(c1, c2, cq, cr)` calculates polynomial  $c_1/c_2$ , storing the quotient polynomial in  $c_q$  and the remainder polynomial in  $c_r$ . For instance, `polydiv((3,5,1), (2,1), cq, cr)` returns  $c_q=(3,1)$  and  $c_r=(-3)$ ; that is,

$$\frac{3 + 5x + x^2}{2 + x} = 3 + x \text{ with a remainder of } -3$$

or

$$3 + 5x + x^2 = (3 + x)(2 + x) - 3$$

`polyroots(c)` find the roots of polynomial  $c$  and returns them in complex row vector (complex even if  $c$  is real). For instance, `polyroots((3,5,1))` returns  $(-4.303+0i, -.697+0i)$  (the roots of  $3 + 5x + x^2$  are  $-4.303$  and  $-.697$ ).

## Syntax

*numeric vector*     `polyeval(numeric rowvector  $c$ , numeric vector  $x$ )`  
*numeric rowvector*   `polysolve(numeric vector  $y$ , numeric vector  $x$ )`  
*numeric rowvector*   `polytrim(numeric vector  $c$ )`  
  
*numeric rowvector*   `polyderiv(numeric rowvector  $c$ , real scalar  $i$ )`  
*numeric rowvector*   `polyinteg(numeric rowvector  $c$ , real scalar  $i$ )`  
  
*numeric rowvector*   `polyadd(numeric rowvector  $c_1$ , numeric rowvector  $c_2$ )`  
*numeric rowvector*   `polymult(numeric rowvector  $c_1$ , numeric rowvector  $c_2$ )`  
*void*                 `polydiv(numeric rowvector  $c_1$ , numeric rowvector  $c_2$ ,  $c_q$ ,  $c_r$ )`  
  
*complex rowvector*   `polyroots(numeric rowvector  $c$ )`

In the above, row vector  $c$  contains the coefficients for a `cols( $c$ ) - 1` degree polynomial. For instance,

$$c = (4, 2, 1)$$

records the polynomial

$$4 + 2x + x^2$$

## Remarks and examples

[stata.com](https://www.stata.com)

Given the real or complex coefficients  $c$  that define an  $n - 1$  degree polynomial in  $x$ , `polyroots( $c$ )` returns the  $n - 1$  roots for which

$$0 = c_1 + c_2x^1 + c_3x^2 + \cdots + c_nx^{n-1}$$

`polyroots( $c$ )` obtains the roots by calculating the eigenvalues of the companion matrix. The  $(n - 1) \times (n - 1)$  companion matrix for the polynomial defined by  $c$  is

$$C = \begin{bmatrix} -c_{n-1}s & -c_{n-2}s & \cdots & -c_2s & -c_1s \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}$$

where  $s = 1/c_n$  if  $c$  is real and

$$s = C \left( \frac{\operatorname{Re}(c_n)}{\operatorname{Re}(c_n)^2 + \operatorname{Im}(c_n)^2}, \frac{-\operatorname{Im}(c_n)}{\operatorname{Re}(c_n)^2 + \operatorname{Im}(c_n)^2} \right)$$

otherwise.

As in all nonsymmetric eigenvalue problems, the returned roots are complex and sorted from largest to smallest, see [M-5] `eigensystem()`.

## Conformability

`polyeval(c, x)`:

*c*:  $1 \times n, n > 0$   
*x*:  $r \times 1$  or  $1 \times c$   
*result*:  $r \times 1$  or  $1 \times c$

`polysolve(y, x)`:

*y*:  $n \times 1$  or  $1 \times n, n \geq 1$   
*x*:  $n \times 1$  or  $1 \times n$   
*result*:  $1 \times k, 1 \leq k \leq n$

`polytrim(c)`:

*c*:  $1 \times n$   
*result*:  $1 \times k, 1 \leq k \leq n$

`polyderiv(c, i)`:

*c*:  $1 \times n, n > 0$   
*i*:  $1 \times 1, i$  may be negative  
*result*:  $1 \times \max(1, n - i)$

`polyinteg(c, i)`:

*c*:  $1 \times n, n > 0$   
*i*:  $1 \times 1, i$  may be negative  
*result*:  $1 \times \max(1, n + i)$

`polyadd(c1, c2)`:

*c*<sub>1</sub>:  $1 \times n_1, n_1 > 0$   
*c*<sub>2</sub>:  $1 \times n_2, n_2 > 0$   
*result*:  $1 \times \max(n_1, n_2)$

`polymult(c1, c2)`:

*c*<sub>1</sub>:  $1 \times n_1, n_1 > 0$   
*c*<sub>2</sub>:  $1 \times n_2, n_2 > 0$   
*result*:  $1 \times n_1 + n_2 - 1$

`polydiv(c1, c2, cq, cr)`:

*input*:  
*c*<sub>1</sub>:  $1 \times n_1, n_1 > 0$   
*c*<sub>2</sub>:  $1 \times n_2, n_2 > 0$   
*output*:  
*c*<sub>q</sub>:  $1 \times k_1, 1 \leq k_1 \leq \max(n_1 - n_2 + 1, 1)$   
*c*<sub>r</sub>:  $1 \times k_2, 1 \leq k_2 \leq \max(n_1 - n_2, 1)$

`polyroots(c)`:

*c*:  $1 \times n, n > 0$   
*result*:  $1 \times k - 1, k = \text{cols}(\text{polytrim}(c))$

## Diagnostics

All functions abort with an error if a polynomial coefficient row vector is void, but they do not necessarily give indicative error messages as to the problem. Polynomial coefficient row vectors may contain missing values.

`polyderiv(c, i)` returns  $c$  when  $i = 0$ . It returns `polyinteg(c, -i)` when  $i < 0$ . It returns `(0)` when  $i$  is missing (think of missing as positive infinity).

`polyinteg(c, i)` returns  $c$  when  $i = 0$ . It returns `polyderiv(c, -i)` when  $i < 0$ . It aborts with error if  $i$  is missing (think of missing as positive infinity).

`polyroots(c)` returns a vector of missing values if any element of  $c$  equals missing.

## Also see

[M-4] [mathematical](#) — Important mathematical functions