scipy.optimize.fsolve¶

scipy.optimize.fsolve(*func*, *x0*, *args=()*, *fprime=None*, *full_output=0*, *col_deriv=0*, *xtol=1.49012e-08*, *maxfev=0*, *band=None*, *epsfcn=None*, *factor=100*, *diag=None***)** **[source]**
(http://github.com/scipy/scipy/blob/v0.14.0/scipy/optimize/minpack.py#L41)

Find the roots of a function.

Return the roots of the (non-linear) equations defined by `func(x) = 0` given a starting estimate.

Parameters:

func *: callable `f(x, *args)`*
A function that takes at least one (possibly vector) argument.

x0 *: ndarray*
The starting estimate for the roots of `func(x) = 0`.

args *: tuple, optional*
Any extra arguments to *func*.

fprime *: callable(x), optional*
A function to compute the Jacobian of *func* with derivatives across the rows. By default, the Jacobian will be estimated.

full_output *: bool, optional*
If True, return optional outputs.

col_deriv *: bool, optional*
Specify whether the Jacobian function computes derivatives down the columns (faster, because there is no transpose operation).

xtol *: float*
The calculation will terminate if the relative error between two consecutive iterates is at most *xtol*.

maxfev *: int, optional*
The maximum number of calls to the function. If zero, then `100*(N+1)` is the maximum where N is the number of elements in *x0*.

band *: tuple, optional*
If set to a two-sequence containing the number of sub- and super-diagonals within the band of the Jacobi matrix, the Jacobi matrix is considered banded (only for `fprime=None`).

epsfcn *: float, optional*
A suitable step length for the forward-difference approximation of the Jacobian (for `fprime=None`). If *epsfcn* is less than the machine precision, it is assumed that the relative errors in the functions are of the order of the machine precision.

factor *: float, optional*
A parameter determining the initial step bound (`factor * || diag * x||`). Should be in the interval `(0.1, 100)`.

diag *: sequence, optional*
N positive entries that serve as a scale factors for the variables.

Returns:

x *: ndarray*

The solution (or the result of the last iteration for an unsuccessful call).

infodict *: dict*

A dictionary of optional outputs with the keys:

`nfev`
: number of function calls

`njev`
: number of Jacobian calls

`fvec`
: function evaluated at the output

`fjac`
: the orthogonal matrix, q, produced by the QR factorization of the final approximate Jacobian matrix, stored column wise

`r`
: upper triangular matrix produced by QR factorization of the same matrix

`qtf`
: the vector `(transpose(q) * fvec)`

ier *: int*

An integer flag. Set to 1 if a solution was found, otherwise refer to *mesg* for more information.

mesg *: str*

If no solution is found, *mesg* details the cause of failure.

See also:

root **(scipy.optimize.root.html#scipy.optimize.root)** Interface to root finding algorithms for multivariate functions.

Notes

`fsolve` is a wrapper around MINPACK's hybrd and hybrj algorithms.

Previous topic

scipy.optimize.root (scipy.optimize.root.html)

Next topic

scipy.optimize.broyden1 (scipy.optimize.broyden1.html)