

# Roots of nonlinear equations

Consider solving a nonlinear equation in one variable  $x$

$$f(x) = 0$$

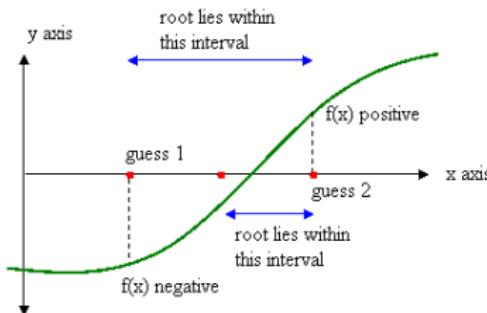
Assume  $f(x)$  smoothly varying, continuous over the range  $[a, b]$ .

A few typical examples of  $f(x)$  are,

$$f(x) = \cos x - x^3, \quad 3x + \sin x - e^x, \quad xe^x - 2, \quad x^3 + 3x - 5 \text{ etc.}$$

If  $x_0$  in the interval  $[a_0, b_0]$  satisfies equation  $f(x_0) = 0$ , then  $x_0$  is a *root* or *zero* of the function and is *one* of the solutions in that interval.

Since  $f(x)$  is continuous and  $[a_0, b_0]$  so chosen such that  $f(a_0)$  and  $f(b_0)$  are of opposite signs, then according to *intermediate value theorem*,  $f(x)$  has at least one root in the interval  $[a_0, b_0]$ .



Finding root numerically starts with guess  $[a_0, b_0]$  – informed or trial-and-error – at which  $f(x)$  has opposite signs.

$[a_0, b_0]$  are said to *bracket* the root.

Iterations proceed by producing a sequence of shrinking intervals  $[a_0, b_0] \rightarrow [a_i, b_i]$  – shrunk intervals always contain one root of  $f(x)$ .

For convergence, necessary to have a good initial guess –  
(i) plotting  $f(x)$  vs  $x$  or (ii) informed expectation.

Algorithms for finding roots of nonlinear equations :

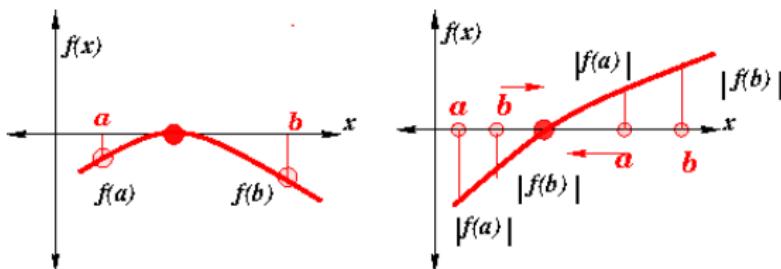
1. Bisection method
2. False position (Regula falsi) method
3. Newton-Raphson method
4. Fixed-point method
5. Laguerre's method (for roots of polynomials)

As is evident, the first four methods are iterative and, therefore, call for user specific precision  $\epsilon$ , typically  $10^{-4}$ .

# Bisection method

Simplest, relatively slower but guaranteed to converge provided the bracketing is done carefully :  $f(a)$  and  $f(b)$  have opposite signs at the interval boundary  $[a, b]$  and  $f(x)$  is continuous.

Things can go wrong – (i) when  $f(x) = 0$  is an extrema i.e. both  $f(a)$ ,  $f(b)$  are always of same sign, or (ii) multiple roots in the interval.



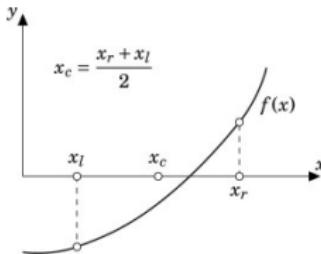
Steps involve in achieving bracketing :

1. Choose  $[a, b]$ , calculate  $f(a)$ ,  $f(b)$ . If  $f(a) * f(b) < 0$ , bracketing done.
2. If  $f(a) * f(b) > 0$  i.e. same sign, check for  $|f(a)| \leq |f(b)|$ .
  - 2.1 If  $|f(a)| < |f(b)|$ , shift  $a$  further left by  $a' = a - \beta(b - a)$
  - 2.2 If  $|f(a)| > |f(b)|$ , shift  $b$  further right by  $b' = b + \beta(b - a)$

## Steps involve in bisection method :

1. Choose bracket  $[a, b]$ , where  $a < b$  and  $f(a) * f(b) < 0$ . If  $|b - a| < \epsilon$ , done! Check also  $f(a)$  and/or  $f(b) < \delta$ .
2. Else bisect the interval  $c = (a + b)/2$ . Choose new interval
  - \*  $[a, c] \equiv [a_1, b_1]$  if  $f(c) * f(a) < 0$
  - \*  $[c, b] \equiv [a_1, b_1]$  if  $f(c) * f(b) < 0$ .Check for  $|b_1 - a_1| < \epsilon$  along with  $f(a_1 \text{ or } b_1) < \delta$ .
3. If not, keep on bisecting the interval and re-adjusting the interval till  $|b_n - a_n| < \epsilon$  along with  $f(a_n \text{ or } b_n) < \delta$  are satisfied.
4. Bisecting  $n$  times, we have a possible solution in the interval length

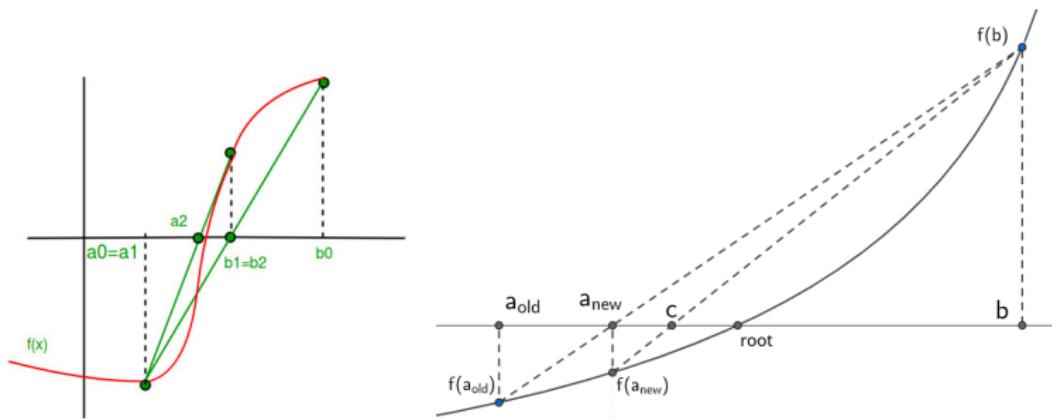
$$\frac{|b_n - a_n|}{2} = \frac{|b - a|}{2^n} \leq \epsilon$$



# Regula falsi method

Interpolation to converge on a root faster than Bisection –

Find slope of the straight line joining  $[a_0, b_0]$  that has bracketed the root. The point  $c_0$  where this straight line crosses  $x - \text{axis}$  is the new  $a$  or  $b$  depending on sign of  $f(c_0)$ .



Unlike Bisection, new interval boundary  $a_n$  or  $b_n$  directly given by  $f(x)$ . Always converges and has improved speed of convergence. Here too, as  $(a, b)$  get close, can lose significant digits.

## Steps involve in Regula falsi :

1. Choose bracket  $[a_0, b_0]$ , where  $a_0 < b_0$  and  $f(a) * f(b) < 0$ . If  $|b_0 - a_0| < \epsilon$ , done! Check also  $f(a_0)$  and/or  $f(b_0) < \delta$ .
2. Calculate slope of the line joining  $f(a_0)$  and  $f(b_0)$  and obtain  $c_0$  where the line crosses abscissa  $y(c_0) = 0$ ,

$$m = \frac{f(b_0) - f(a_0)}{b_0 - a_0} = \frac{f(b_0) - f(c_0)}{b_0 - c_0} \Rightarrow c_0 = b_0 - \frac{(b_0 - a_0) * f(b_0)}{f(b_0) - f(a_0)}$$

Reference point can as well be  $f(a_0)$ . If  $f(x)$  is convex or concave, then one of the points  $a, b$  is fixed and the other varies with iterations. After  $n$ -th step,

$$c_n = b_n - \frac{(b_n - a_n) * f(b_n)}{f(b_n) - f(a_n)}$$

3. If  $f(a_n) * f(c_n) < 0$ , then root lies to left of  $c_n \Rightarrow b_{n+1} = c_n$  and  $a_{n+1} = a_n$ . If  $|c_{n-1} - c_n| < \epsilon$  then  $c_n$  is the root  $f(c_n) \approx 0$ .
4. If  $f(b_n) * f(c_n) < 0$ , then root lies to right of  $c_n \Rightarrow a_{n+1} = c_n$  and  $b_{n+1} = b_n$ . If  $|c_{n-1} - c_n| < \epsilon$  then  $c_n$  is the root  $f(c_n) \approx 0$ .

# Fixed Point method

Finding root of  $f(x) = 0$  amounts to finding  $x_0$  where  $f(x_0) = 0$ .

★ Fixed point of a function  $g(x)$  is  $x = g(x)$ .

In fixed point method,  $f(x) = 0$  is replaced by  $x = g(x)$ .

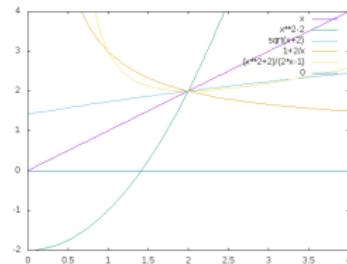
★ If  $x_n$  is a fixed point of  $g(x)$ , then  $x_n$  is a root of  $f(x)$ .

The iteration scheme is, starting from initial guess  $x_1$ , then  $x_2, x_3, \dots$

$$x_{n+1} = g(x_n) \Rightarrow x_{n+1} \approx g(x_n) \text{ i.e. } |x_{n+1} - x_n| < \epsilon$$

For a given  $f(x) = 0$ , there can be many equivalent fixed point problems with different choices for  $g(x)$ . For example,

$$f(x) = x^2 - x - 2 = 0 \Rightarrow x = g(x) = x^2 - 2, \sqrt{x+2}, 1 + \frac{2}{x}, \frac{x^2 + 2}{2x - 1}, \dots$$



Not all choices converge equally! Choices may introduce unwanted singularities!! Solution is not guaranteed!!!

## Newton-Raphson method

Involves both  $f(x)$  and  $f'(x)$  but does not require bracketing. Method is based on Taylor's expansion.

Works for multivariate functions. Converges quadratically  $\Rightarrow$  near root the number of significant digits approximately doubles with each step.

To solve  $f(x) = 0$ , Taylor expand  $f(x)$  around initial guess  $x_0$ ,

$$f(x) = f(x_0) + (x - x_0)f'(x_0) + \frac{1}{2!}(x - x_0)^2f''(x_0) + \dots$$

If closer to root  $(x - x_0)^2 \approx 0$ , we stop at  $f'(x)$  term,

$$f(x) = f(x_0) + (x - x_0)f'(x_0) = 0 \Rightarrow x = x_0 - \frac{f(x_0)}{f'(x_0)}$$

then  $x$  is better approximation of root than  $x_0$  but involves taking derivative. Useful if derivative is cheaper to evaluate and hence to code.

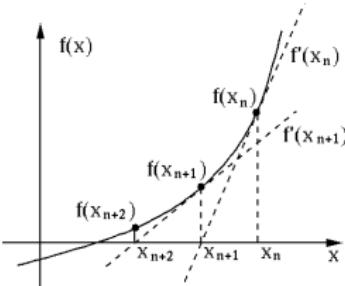
Approximation to the root can be improved iteratively to move from the  $x_0$  towards the root,

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n = 1, 2, \dots$$

Alternatively, use finite difference to approximate derivative – a variant of Newton's method called Secant method,

$$f'(x) \approx \frac{1}{2h} [f(x+h) - f(x-h)] + O(h^3)$$

Finite difference requires two initial guesses  $x_0, x_1$  corresponding to  $x \pm h$ .



The steps of Newton-Raphson method are

1. Make a good guess of  $x_0$ .
2. Evaluate  $f(x)$  and  $f'(x)$  at  $x = x_0$ .
3. Use iterative updating  $x_n \rightarrow x_{n+1}$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n = 1, 2, \dots$$

4. Continue to improve the estimate of the root until  $|x_{n+1} - x_n| < \epsilon$  and/or  $f(x_n) \approx \delta$ .

# Multivariable system

Both **Fixed point** and **Newton-Raphson** can be employed for solving multivariable nonlinear system of equations.

$$f(\mathbf{x}) = \begin{pmatrix} f_1(x_1, x_2, \dots, x_n) \\ f_2(x_1, x_2, \dots, x_n) \\ \vdots \\ f_n(x_1, x_2, \dots, x_n) \end{pmatrix} = \mathbf{0} \quad \text{where } \mathbf{x}^T = (x_1, x_2, \dots, x_n)$$

No concept of bracketing in higher dimension  $\Rightarrow$  guaranteed converging methods like **Bisection** does not exist!

The **fixed point functions**  $\mathbf{g}(\mathbf{x})$  can be written as

$$\left. \begin{array}{lcl} x_1 & = & g_1(x_1, x_2, \dots, x_n) \\ x_2 & = & g_2(x_1, x_2, \dots, x_n) \\ \vdots & & \\ x_n & = & g_n(x_1, x_2, \dots, x_n) \end{array} \right\} \Rightarrow \mathbf{x} = \mathbf{g}(\mathbf{x}) \Rightarrow \mathbf{x}_{k+1} = \mathbf{g}(\mathbf{x}_k)$$

The stopping criteria would be,

$$\frac{\|\mathbf{x}_{k+1} - \mathbf{x}_k\|}{\|\mathbf{x}_{k+1}\|} < \epsilon \quad \text{where, } \|\mathbf{x}_k\|^2 = (x_1)_k^2 + (x_2)_k^2 + \dots + (x_n)_k^2$$

For example, try solving the following

$$x_1^2 + x_1 x_2 = 10 \quad \text{and} \quad x_2 + 3x_1 x_2^2 = 57$$

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \mathbf{g}(\mathbf{x}) = \begin{pmatrix} g_1(x_1, x_2) \\ g_2(x_1, x_2) \end{pmatrix} = \begin{pmatrix} \frac{10 - x_1^2}{x_2} \\ 57 - 3x_1 x_2^2 \end{pmatrix}$$

Using an initial guess of  $x_1 = 1.5$ ,  $x_2 = 3.5$  yields divergence!

$$x_1 = \frac{10 - (1.5)^2}{3.5} = 2.21429, \quad x_2 = 57 - 3(2.21429)(3.5)^2 = -24.37516$$

$$x_1 = \frac{10 - (2.21429)^2}{-24.37516} = -0.20910, \quad x_2 = 57 - 3(-0.20910)(-24.37516)^2 = 429.709$$

Continuing the iteration shows it is diverging in  $x_2$ . A different choice of  $\mathbf{g}(\mathbf{x})$  works rather well,

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \sqrt{(10 - x_1 x_2)} \\ \sqrt{(57 - x_2)/3x_1} \end{pmatrix}$$

$x_1$	$x_2$	$x'_1$	$x'_2$	$\epsilon$
1.5000	3.5000	2.1794	2.8605	0.259459
2.1794	2.8605	1.9405	3.0496	0.084286
1.9405	3.0496	2.0205	2.9834	0.028792
⋮				
1.9999	3.0001	2.0000	3.0000	0.000046

Taylor expansion in **Newton-Raphson** for multivariable function, say  $f_1(x_1, x_2, \dots, x_n)$  of  $\mathbf{f}(\mathbf{x})$  in terms of solutions after *i-th* iteration

$$f_1\left(x_1^{(i+1)}, x_2^{(i+1)}, \dots, x_n^{(i+1)}\right) \approx f_1\left(x_1^{(i)}, x_2^{(i)}, \dots, x_n^{(i)}\right) + \frac{\partial f_1}{\partial x_1}\Big|_{x^{(i)}}\left(x_1^{(i+1)} - x_1^{(i)}\right) + \\ \frac{\partial f_1}{\partial x_2}\Big|_{x^{(i)}}\left(x_2^{(i+1)} - x_2^{(i)}\right) + \dots + \frac{\partial f_1}{\partial x_n}\Big|_{x^{(i)}}\left(x_n^{(i+1)} - x_n^{(i)}\right)$$

Similar Taylor expansion for  $f_2, f_3, \dots, f_n$  yields,

$$\mathbf{f}(\mathbf{x}^{(i+1)}) = \mathbf{f}(\mathbf{x}^{(i)}) + \mathbf{J}(\mathbf{x}^{(i)}) \left(\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)}\right) = 0 \Rightarrow \mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \mathbf{J}^{-1}(\mathbf{x}^{(i)}) \mathbf{f}(\mathbf{x}^{(i)})$$

where  $\mathbf{J}(\mathbf{x}^{(i)})$  is the Jacobian matrix

$$\left[\mathbf{J}(\mathbf{x}^{(i)})\right]_{pq} = \frac{\partial f_p(\mathbf{x})}{\partial x_q}\Big|_{\mathbf{x}^{(i)}} = \begin{bmatrix} \frac{\partial f_1(\mathbf{x}^{(i)})}{\partial x_1} & \frac{\partial f_1(\mathbf{x}^{(i)})}{\partial x_2} & \dots & \frac{\partial f_1(\mathbf{x}^{(i)})}{\partial x_n} \\ & \vdots & & \\ \frac{\partial f_n(\mathbf{x}^{(i)})}{\partial x_1} & \frac{\partial f_n(\mathbf{x}^{(i)})}{\partial x_2} & \dots & \frac{\partial f_n(\mathbf{x}^{(i)})}{\partial x_n} \end{bmatrix}$$

Compare multivariable and single-variable Newton-Raphson,

$$\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \left(\mathbf{f}'(\mathbf{x}^{(i)})\right)^{-1} \mathbf{f}(\mathbf{x}^{(i)}) \text{ and } \mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \mathbf{J}^{-1}(\mathbf{x}^{(i)}) \mathbf{f}(\mathbf{x}^{(i)})$$

Consider the previous example

$$\begin{aligned} x_1^2 + x_1 x_2 - 10 &= 0 \\ x_2 + 3x_1 x_2^2 - 57 &= 0 \end{aligned} \quad \left. \right\} \quad J = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2x_1 + x_2 & x_1 \\ 3x_2^2 & 1 + 6x_1 x_2 \end{bmatrix}$$

For initial guess, say  $x_1^{(0)} = 1$ ,  $x_2^{(0)} = 1$ ,

$$f(x^{(0)}) = \begin{pmatrix} f_1(x_1^{(0)}, x_2^{(0)}) \\ f_2(x_1^{(0)}, x_2^{(0)}) \end{pmatrix} = \begin{pmatrix} -8 \\ -53 \end{pmatrix} \Rightarrow J = \begin{bmatrix} 3 & 1 \\ 3 & 7 \end{bmatrix} \rightarrow J^{-1} = \frac{1}{18} \begin{bmatrix} 7 & -1 \\ -3 & 3 \end{bmatrix}$$

Hence the solution after first iteration is,

$$\begin{pmatrix} x_1^{(1)} \\ x_2^{(1)} \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} - \frac{1}{18} \begin{bmatrix} 7 & -1 \\ -3 & 3 \end{bmatrix} \begin{pmatrix} -8 \\ -53 \end{pmatrix} = \begin{pmatrix} 1.1667 \\ 8.5000 \end{pmatrix}$$

$n$	$x_1$	$x_2$
1	1.1667	8.5000
2	1.5670	3.6878
3	2.0108	2.8824
4	1.9992	3.0023
5	2.0000	3.0000

Converges quadratically but cost of computing Jacobian is  $N^2$  for function evaluations and  $N^3$  for solving linear equations.

# Laguerre's method

A polynomial of degree  $n$  has exactly  $n$  roots  $\alpha_i$  ( $i = 1, 2, \dots, n$ )

$$P(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \cdots + a_nx^n = (x - \alpha_1)(x - \alpha_2) \cdots (x - \alpha_n)$$

Polynomials with real  $a_i$ , roots can be real or complex conjugate pair.  
We will restrict ourselves to only real roots.

A two step method –

(i) Laguerre's method followed by (ii) deflating the polynomial.

Using Laguerre's method determine the root  $\alpha_1$  of  $P(x)$ , then obtain a reduced polynomial  $Q(x)$  of degree one less than  $P(x)$ .

$$P(x) = (x - \alpha_1) Q(x)$$

$$Q(x) = (x - \alpha_2) R(x) \Rightarrow P(x) = (x - \alpha_1)(x - \alpha_2) R(x) \text{ etc.}$$

where the roots of  $Q(x)$  are the remaining roots of  $P(x)$ . In each of  $n$  steps, the Laguerre determines the roots  $\alpha_i$  while deflation determines the remainder polynomial.

Method used for deflation is synthetic division method (learnt in class IX).

## Laguerre algorithm proceeds as

1. Begin with an initial guess  $\beta_0$ .
2. If  $\beta_0$  is bang on one of the roots, i.e.  $P(\beta_0) \approx 0$ , go for deflation.
3. Else calculate the following

$$G = \frac{P'(\beta_k)}{P(\beta_k)}, \quad H = G^2 - \frac{P''(\beta_k)}{P(\beta_k)}$$
$$\Rightarrow a = \frac{n}{G \pm \sqrt{(n-1)(nH - G^2)}}$$

Choose the sign in the denominator of  $a$  such as to give the denominator the larger absolute value.

4. Set  $\beta_{k+1} = \beta_k - a$  as new trial.
5. Iterate till  $|\beta_{k+1} - \beta_k| < \epsilon$  and set  $\alpha_1 = \beta_k$ . Check  $P(\alpha_1) \approx 0$ .
6. Go for deflation to reduce the degree of the polynomial and do the above iteration all over again to find  $\alpha_2$  and so on.

Next deflation : divide  $P(x)$  by  $(x - \alpha_1)$  to get  $Q(x)$

$$\frac{P(x)}{x - \alpha_1} = Q(x), \text{ followed by } \frac{Q(x)}{x - \alpha_2} = R(x) \text{ etc.}$$

## Deflation by Synthetic division method :

- ★ Arrange terms in  $P(x)$ , the dividend, in descending order of power, 0 (zero) for missing power. Consider polynomial  $-x^3 + 3x^2 - 4$

$$\frac{P(x)}{x - \alpha_1} = \frac{-x^3 + 3x^2 - 4}{x - 2} \Rightarrow \text{divisor} = 2, \text{ coeffs} = [-1, 3, 0, -4]$$

- ★ Write the divisor and coefficients in the manner given below. Bring down the first or leading coefficient i.e.  $-1$  below the horizontal line.
- ★ Multiply the coefficient of leading power  $-1$  with the divisor  $2 \times -1 = -2$  and add it to the coefficient of the next lower power  $3 + (-2) = 1$  and bring it down below the horizontal line again.
- ★ Multiply it again with divisor and continue this process till the end.

	-1	3	0	-4
2	+	-2	2	4
	-1	1	2	0

- ★ If  $\alpha_1 = 2$  is a root then the last sum i.e. the last entry below the horizontal line, which gives the remainder, must be zero.

- The reduced lower degree polynomial  $Q(x)$  has the numbers below the horizontal line as its coefficients

$$\frac{P(x)}{x - \alpha_1} = \frac{-x^3 + 3x^2 - 4}{x - 2} = -x^2 + x + 2 = Q(x)$$

- Repeat the above process with  $Q(x)$  and keep doing for successive roots till you get the final monomial  $(x - \alpha_n)$ .

$$\begin{array}{c|ccc} & -1 & 1 & 2 \\ 2 & + & -2 & -2 \\ \hline & -1 & -1 & 0 \end{array} \Rightarrow R(x) = -x - 1$$

The polynomial in the example is thus factorized in terms of its roots as

$$P(x) = -x^3 + 3x^2 - 4 = (x - 2)(x - 2)(-x - 1)$$

Find all the roots of the following polynomial

$$P(x) = 6x^3 - 11x^2 - 26x + 15 \quad \text{answer : } x = 3, -5/3, 1/2$$

$$P(x) = x^4 - x^3 - 7x^2 + x + 6 \quad \text{answer : } x = 1, -2, 3, -1$$