

[Help](#)

```
#include      <math.h>
#include      <stdlib.h>
#include      "cdo_math.h"
#include "pnl/pnl_cdf.h"

/**
 * valuation of a polynom P in a point (be careful : the
 * polynom has non constant term P(0)=0)
 *
 * @param degree : degree of the polynom
 * @param a : array containing the coefficients of the
 * polynom (be careful : a[0] corresponds to the coefficient
 * in front of x^1)
 * @param x : scalar
 * @return P(x)
 */

double      evaluate_poly(int degree, const double *a, double x)
{
    double      result;
    int          n;

    if (degree == 0) result = 0.;
    else
    {
        result = a[degree - 1] * x;
        for (n = degree - 2; n >= 0; n--)
            result = (result + a[n]) * x;
    }

    return (result);
}

/**
 * valuation of the derivative of a polynom P in a point (be careful : the
 * polynom has non constant term P(0)=0)
 *
 * @param degree : degree of the polynom
 * @param a : array containing the coefficients of the
```

```
* polynom (be careful : a[0] corresponds to the coefficient
* in front of x^1)
* @param x : scalar
* @return (P')(x)
*/

double      evaluate_dpoly(int degree, const double *a, double x)
{
    double      result;
    int          n;

    if (degree == 0) result = 0.;
    else
    {
        result = degree * a[degree - 1];
        for (n = degree - 2; n >= 0; n--)
            result = (n + 1.) * a[n] + result * x;
    }
    return (result);
}
```