Implementation of error function using integral representation

D.V Fedorov
Aarhus University

The error function and the complimentary error function are implemented using direct integration of their integral representations with GSL's Gauss-Kronrod integration rutines.

I. INTRODUCTION

The error function, $\mathrm{erf}(x)$, and the complementary error function, $\mathrm{erfc}(x)$, are defined as [1]

$$\mathrm{erf}(x) \doteq \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \,, \qquad (1)$$

$$\mathrm{erfc}(x) \doteq \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2} dt = 1 - \mathrm{erf}(x) \,. \qquad (2)$$

They have the following reflection properties,

$$\mathrm{erf}(-x) = -\mathrm{erf}(x) \,, \qquad (3)$$

$$\mathrm{erfc}(-x) = 2 - \mathrm{erfc}(x) \,. \qquad (4)$$

The error function appears—among many other things—in the electrostatic potential of a spherically symmetric Gaussian charge disribution.

This report describes the implementation of the error and the complimentary error functions by direct integration of their integral representations (3) and (4). Section II describes the details of the implementation, section III illustrates the results, section IV presents the conclusions, and the appendix lists the source code of the implemented functions.

II. IMPLEMENTATION

First, for negative x the reflection formulae (3) and (4) can be used to switch the argument to positive values.

The definition (1) of the $\mathrm{erf}(x)$ function represents an integral over a smooth integrand which can be calculated using the high precision `gsl_integration_qag` routine with 61-points as long a x is not very large.

For large x one needs to switch to the definition (2) of the $\mathrm{erfc}(x)$ function and use the `gsl_integration_qagiu` routine which calculates integrals over a semi-infitine interval.

Experiments show that the optimal point to switch from (1) to (2) is around $x = 3$. Then absolute accuracy of about 10^{-14} is reached with at most 75 evaluations [2] of the integrand for any $x \in [-\infty, +\infty]$.

III. RESULTS

Figure 1 shows the implemented `myerf` function compared to the `erf` function from the C standard library. The figure was made with `pyxplot` using the `pdf` terminal. It is included via the `graphicx` package.

The implemented complementary error function `myerfc` is illustrated on figure 2 together with the `erfc` function from the C standard library. The figure was made with `gnuplot` using the `latex` terminal and was directly inlcluded using the `\intput{plot2.tex}` command.

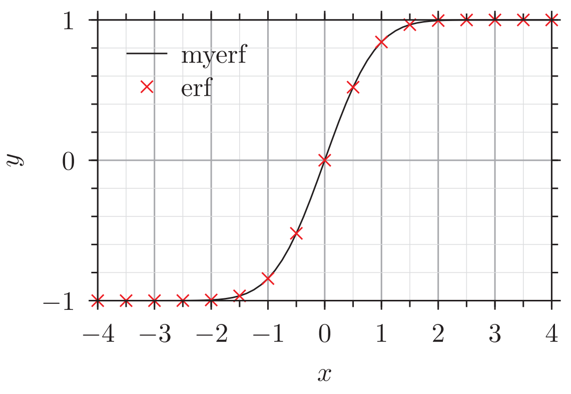

FIG. 1: The implemented error function `myerf(x)` together with the `erf(x)` function from C standard library.

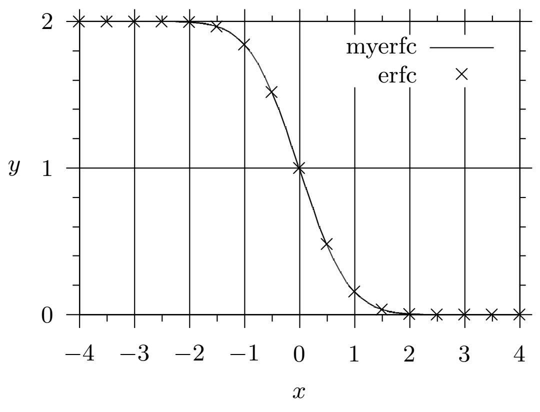

FIG. 2: The implemented complimentary error function `myerfc(x)` together with the `erfc(x)` function from the standard libraty.

IV. CONCLUSION

The error function and the complimentary error function are implemented by direct numerical calculation of the defining integrals using the numerical integration routines `gsl_integration_qag` and `gsl_integration_qagiu`. The absolute error of 10^{-14} is achieved for any argument with at most 75 evaluations of the integrands.

V. APPENDIX

```
#include <stdio.h>
#include <math.h>
#include <gsl/gsl_integration.h>
#define MID 3
#define ACC 1e-14

double myerf(double x);
double myerfc(double x);

int calls;

double myerf(double x){
    if(x<0)return -myerf(-x);
    if(x>=MID) return 1-myerfc(x);

    int limit=1000;
    gsl_integration_workspace* wspace = gsl_integration_workspace_alloc (limit);

    double integrand(double t, void* p){ calls++; return exp(-t*t);}
    gsl_function F = {.function=integrand, .params=NULL};

    double result,error,eps=0;
    calls=0;
    gsl_integration_qag(&F,0,x,ACC,eps,limit,GSL_INTEG_GAUSS61,wspace,&result,&error);
    fprintf(stderr,"x=%14g, err=%14g, calls=%4i\n",x,error,calls);

gsl_integration_workspace_free (wspace);
return result*2/sqrt(M_PI);
}

double myerfc(double x){
    if(x<0)return 2-myerfc(-x);
    if(x<MID)return 1-myerf(x);

    int limit=1000;
    gsl_integration_workspace* wspace = gsl_integration_workspace_alloc(limit);

    double integrand(double t, void* p){ calls++; return exp(-t*t);}
    gsl_function F = {.function=integrand, .params=NULL};

    double result,error,eps=0;
    calls=0;
    gsl_integration_qagiu(&F,x,ACC,eps,limit,wspace,&result,&error);
    fprintf(stderr,"x=%14g, err=%14g, calls=%4i\n",x,error,calls);

gsl_integration_workspace_free (wspace);
return result*2/sqrt(M_PI);
}
```

[1] Andrews, Larry C.; Special functions of mathematics for engineers.
[2] The log file from the archive of the project.