# Lab #5

Computational Physics (**Phys381**)
R. Ouyed
**Due March 25, 2014 (at the end of class)**

- The latexed report is worth **20%** of the total mark. Only well documented and neatly presented reports will be worth that much! It is not sufficient to give only numerical results and show plots, you should also discuss your results. Include complete figure captions, introduction and conclusion sections.
- Your report must be in a *two-column format.*
- Your report should include your *Fortran code* and *Gnuplot scripts* in an Appendix using the *verbatim* command.
- If applicable, animations should be shown to the teacher and/or TA before handing in the lab report. The animation should be well documented and contains necessary information (student name, assignment number, run time etc ...). Basically, the information should be included in each frame before they are put together.
- You must name your report using the names (last names only): *student1-student2-phys381-lab#.pdf.*
- **Procedure for Handing in your lab report (see instructions on phys381 wbsite)**:

  1) Set permission to your PDF report as 644. It means:
  chmod 644 *student1-student2-phys381-lab#.pdf*
  2) cp -a *student1-student2-phys381-lab#.pdf* **/home/ambrish/phys381/labs/lab#**
  3) Copy a second time to ensure that your exam copied correctly. If you are prompted as to whether or not you would like to replace the existing file, then you report has been successfully submitted.
- **You must check with your TAs (Ambrish or Zach) that your report was received and is readable BEFORE you leave the lab.**

**→ This is the most challenging laboratory of phys381. Use you time wisely and optimize the tasks with you lab partner.**

# 1 Truncated Error Function (Total: 80 Marks)

Fortran 90 has a built-in function to evaluate the error function; it is called **erf()**. The goal here is to compare this intrinsic function to the one based on the truncated series below. You should start this laboratory by first reading sections 3.1.3 and 3.2.1 in Ouyed&Dobler textbook.

The error function erf(x) is defined by an integral:

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-t^2)dt \ .$$

It is known that for **small** $x$ **(i.e.** $x << 1$**)**, a good and efficient way to evaluate the $erfx$ is to use the truncated series:

$$erf(x) = \frac{2x}{\sqrt{\pi}} \sum_{k=0}^{\infty} \frac{(-1)^k}{k!(2k+1)} x^{2k} \simeq \frac{2x}{\sqrt{\pi}} \sum_{k=0}^{N-1} \frac{(-1)^k}{k!(2k+1)} x^{2k} \ ,$$

where $N$ is number of terms in the series estimate of the function.

i) **[20 Marks]** Write a Fortran code to evaluate $erf(x)$ using the truncated series prescription given above. Name your code " *phys381erf.f90* " and it should contain:

(a) a Horner Algorithm to perform the series calculations [*you will have to adapt the Horner's scheme since the series is in* $x^{2k}$ *powers not in* $x^k$]; see section 3.1.2 in Ouyed&Dobler.

(b) a subroutine which performs the factorial function (see appendix A below).

(c) Furthermore, your code should make a call to Fortran's intrinsic error function.

(ii) **[10 Marks]** Using your Fortran code, estimate the $erf(x)$ for $-0.01 \leq x \leq 0.01$ and for $N = 3, 10, 30, 100$. Save your results into one single file (name it "*erfout.data*"). The output format must contain the following 6 columns:

```
x,  erf(x)@f90,  erf(x)@N=3,  erf(x)@N=10,  erf(x)@N=30,  erf(x)@N=100
```

where $erf(x)@f90$ is the error function intrinsic to Fortran 90.

Plot the five corresponding curves (erf(x) versus x) in one panel. In the second panel of your figure plot the four curves representing [erf(x)@N - erf(x)@f90]; meaning the difference between the series estimate of the function and the estimate from the Fortran intrinsic function.

(iii) **[10 Marks]** Comment on the minimum value of $N$ necessary to trust your truncated series prescription of the error function?

(iv) **[10 Marks]** Repeat the above for $-0.1 \leq x \leq 0.1$. How does Fortran 90 handles the error function for larger values of $x$?

### 1.1 The complementary error function $erfc(x)$

We now consider the complementary error function $erfc(x)$ (**which applies for** $x > 1$), defined by

$$erfc(x) = 1 - erf(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty \exp(-t^2) dt \ .$$

The *asymptotic expansion* for $erfc(x)$ is (i.e. for $x > 1$)

$$erfc(x) = \frac{e^{-x^2}}{x\sqrt{\pi}} \sum_{N=0}^{\infty} (-1)^N \frac{(2N)!}{N!(2x)^{2N}}$$

(i) **[10 Marks]** Write a Fortran code to calculate erfc(x) using the prescription given above. This code should be separate from the one you developed earlier. *Hint*: make a copy of the first code and adapt it (by making necessary changes) to the erfc(x).

(ii) **[10 Marks]** Here you are asked to estimate the complementary error function $erfc(x)$ at $x = 2.5$ using the series expansion for $N$ up to 20. Basically, you need to plot $erfc(2.5)$ versus $N$ and compare it to the $erfc(2.5)$ value using the Fortran intrinsic function (erfc(2.5)@f90).

(iii) **[10 Marks]** In an *inlet* – a small window inside your figure (see Appendix B below) – plot [erfc(2.5)@N - erfc(2.5)@f90] for $4 \leq N \leq 8$.

Looking at the inlet, at what value of $N$ do you get the best estimate of the complementary error function (i.e. at what value of $N$ is [erfc(2.5)@N - erfc(2.5)@f90] minimal)?

## 2 Error Function in the literature (10 Bonus Marks)

As part of your Discussion&Conclusion section you are asked to add a few paragraphs (see questions below) about efforts in the literature at improving computational estimates of the error function. Your report should have the reference below included in the list of references.

Go to the phys381 **pjl.ucalgary.ca** site and download the following paper (it is next to Lab#5 link):

```
Title: Efficient Computation of erfc(x) for Large Arguments
Authors: C. Tellambura,  and A. Annamalai
```

Describe what is performed in this paper and how it compares to what you have learned from this part I of this laboratory. In particular how does the series description of the error function compare to the one we used in this laboratory? In your opinion, what is the main conclusion of the paper?

## A Factorial function

Function to calculate factorials recursively. Also note the RESULT value in the function. This syntax is needed so that the subprogram knows when (and what) to return to the main program when the recursive loop is finished.

```
recursive function factorial(n) result(nfact)
implicit none
integer, intent(in) :: n
double precision :: nfact ! integer would quickly overflow
if (n > 0) then
nfact = n * factorial(n-1)
else
nfact = 1
endif
endfunction factorial
```

## B Inlets in Gnuplot

An inlet is a small figure inside the main figure.

*Hint:* First draw a main graph in the multiplot mode. Then, move the origin to a vacant place in the main figure where you want the inlet placed, and draw a small graph there. The X and Y ranges should be determined to enlarge the place where you want to magnify. The small figure is the same as the main one except for the ranges, so that you can use the replot command.

The example below is an inlet used to **zoom in** into a region in the main graph:

```
set size 1,1
set origin 0,0
set multiplot
set xrange [0:26]
.
. . main plot here "plot ...."
.
unset key
unset xlabel
unset ylabel
unset title
set xrange [4:8]
set origin 0.15,0.15
set size 0.45,0.45
replot
```