Homework Assignments

All homework assignments (reports + programs) have to be submitted by e-mail before the deadlines (have physics 420 in the subject line). A carbon copy of the report has to be in the instructor's mailbox (department of physics) or given in class before the due day and time.

All submitted programs have to have a good description and be self-explanatory as possible. Example

```
/* ------------------------------------------------------------
  John Doe               Physics 420          February xx, 2010
  Assignment # hw xx
  Integration by trapezoid rule of f(x) on [a,b]
  Method: trapezoid rule
  ------------------------------------------------------------
  Input:
    f - Function to integrate
    a - Lower limit of integration
    b - Upper limit of integration
    n - number of intervals
  Output:
    r - Result of integration
  Compiler: Dev C++
  Comments:
  ------------------------------------------------------------*/
```

Homework 0.1: Warming up (Due on Wednesday, January 20, 2010 by 13:30)

1 In classes that you took earlier you met problems that were difficult, or impossible to solve analytically. Suggest three problems for solving with a computer. Explain why do you need computational physics for solving these problems. Later we may use these problems for midterm projects.

2 Install a C/C++ compiler or Fortran, or Java on your computer (if you don't have one). Remember that later you may select a different compiler. See the list of available compilers on the web page. http://www.odu.edu/~agodunov/computing/lib_soft.html

Homework 0.2: Simple programs (Due on Wednesday, January 27, 2010 by 13:30)

1 Write a program to solve the quadratic equation $ax^2 + bx + c = 0$ by using the quadratic formula to obtain roots. Your program should also be capable to handle complex roots.

2 Write a program that calculates a series of Fibonacci numbers and checks which ones are prime numbers. Test your program on first 25 Fibonacci numbers.

3 Write a program that calculates the values of the Legendre polynomials using the recurrence relation $(n+1)P_{n+1}(x) - (2n+1)xP_n(x) + nP_{n-1}(x) = 0$ where $P_0(x) = 1$ and $P_1(x) = x$ Test your program on available analytic solutions. What is the largest *n* when you can still use the recurrence relation? (a very helpful reference: "Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables" by Abramowitz, M. and Stegun, I.A.)

Homework 1: Roots of nonlinear equations (Due on Wednesday, February 3, 2010 by 13:30)

1 Write a program that solves a nonlinear equation f(x)=0 using one of the closed (bisectional or false position) or one of the open domain methods (Newton's or secants). The program should have an upper limit on the number of iterations.

2 Using your program solve the following equations.

a) $f(x) = x + \cos(x) = 0$ on [-10.0, +10.0]

b) $f(x) = \exp(x) - x\sin\left(\frac{\pi x}{2}\right) = 0$ on [-5.0, +2.5]

c) $f(x) = x^3 - 2x^2 - 2x + 1 = 0$ on [0.0, +2.0]

Report how many iterations does it take to get a tolerance of 1.0e-6. Does the number of iterations depend on the initial guess?

3 Modify your program to find all real roots of the following equations using the brute force approach

a) $x^4 - 6x^3 + 12x^2 - 36x - 18 = 0$ on [-10.0, +10.0]

b) $f(x) = \exp(x) - x\sin\left(\frac{\pi x}{2}\right) = 0$ on [-10.0, +10.0].

4 Consider a one-dimension quantized system, namely a finite square well with $U(x)=-U_0$ for $|x|<a$, and $U(x)=0$ for $|x|>a$. Solutions can be found from the following nonlinear equations

for even states $\sqrt{U_0 - |E|}\tan\sqrt{2m(U_0 - |E|)\,a^2/\hbar^2} = \sqrt{|E|}$

for odd states $\sqrt{U_0 - |E|}\cot\sqrt{2m(U_0 - |E|)\,a^2/\hbar^2} = -\sqrt{|E|}$

In textbooks these equations are solved graphically. Find numerically solutions for a few lowest bound states using closed or open domain methods for $U_0 = 10,\ m = 1,\ a = 1,\ \hbar = 1$ (here we use atomic units).

Homework 2: Ordinary differential equations (Due on Monday, February 22, 2010 by 13:30)

1 Using Taylor series approach derive following equations for numerical differentiation f(x) at point x_i (the question is related to numerical differentiation)

$$f_x = \frac{f_{i+1} - f_{i-1}}{2\Delta x} - \frac{1}{6} f_{xxx}(\zeta)\Delta x^2$$

$$f_{xx} = \frac{f_{i+1} - 2f_i + f_{i-1}}{\Delta x^2} - \frac{1}{12} f_{xxxx}(\zeta)\Delta x^2$$

2 Write a program that numerically solves an initial value problem for a first order ODE using the 4-th order Runge-Kutta method (and some other methods if you want).

3 Solve numerically following ODEs

$$a)\quad \frac{dx}{dt} + x = 0, \quad x(0) = 1.0, \qquad t = 0.0 \to 2.0$$

$$b)\quad \frac{dx}{dt} + x\cos(t) - \frac{1}{2}\sin(2x) = 0, \quad x(0) = 1.0, \qquad t = 0.0 \to 10.0$$

$$c)\quad \frac{dx}{dt} - tx^2 = 0, \quad x(0) = 1.0, \qquad t = 0.0 \to 1.0$$

Study effects of the step size and compare results of different methods if you used more than one. Compare with analytic solutions, if you can solve analytically any on the equations above.

4 Population growth of many species can be modeled as

$$\frac{dN}{dt} = aN - bN^2 \qquad N(0) = N_0$$

where N is the population, *aN* represents the birthrate, and bN^2 represents the death rate due to various causes (limited resources, diseases, other negative factors). If N(0)=100, a=0.1, and b=0.0002, calculate N(t) for t=0.0 to 100.0 years

5 Consider a predator-prey model with rabbits and foxes. The following system of differential equations (Lotka-Volterra model) describes the change in populations of rabbits (r) and foxes (f)

$$\frac{dr}{dt} = k_r r - k_{fr} fr$$

$$\frac{df}{dt} = -k_f f + k_{rf} rf$$

where k_r is the prey birth rate, k_{fr} is prey death proportionality constant, k_f is the predator death rate, and k_{rf} is the predator birth fraction. If r(0)=100 and f(0)=15, k_r=2.0, k_f=1.0, k_{fr}=0.02, k_{rf}=0.01, calculate the population of rabbits and foxes as a function of time from t=0.0 till t=16.0.

You may write your own program for solving a system of first order ODEs, or use one from the course web page (system of N first order equations (4th order Runge-Kutta), program ode_n1.cpp)

Homework 3: Numerical integration (Due on Friday, March 19, 2010 by 21:00)

1 Derive the Newton-Cotes formulas for polynomial of degree n=4 (and, optional, for n=5)

2 Write a program that calculates an integral with a given integrand f(x) in the region [a, b] by one of the Newton-Cotes rules with n>1 (Simpson's 1/3 rule, or Simpson's 3/8, or Milne, or higher). (examples from the lecture notes or from the C++/Fortran library page may help).

3 Evaluate numerically the following integrals

a) $\int_0^1 \sin(\pi x)dx$

b) $\int_0^1 \frac{1}{1-0.998x^2}dx$

c) $\int_0^{2\pi} x\sin(30x)\cos(50x)dx$

d) $\int_0^1 \frac{x}{e^x-1}dx$

using your code with various numbers of intervals. Study the effect of the number of intervals on the results. Report, if your program fails to compute some of the integrals. Explain why.

4 Evaluate numerically following improper integrals.

a) $\int_0^\infty \frac{\exp(-x^2)}{1+x^2}dx$

b) $\int_0^\infty \frac{x\sin(x)}{1+x^2}dx$

There are multiple ways to evaluate improper integrals. Explain your choice.

5 Optional. Getting experience with quanc8.cpp or quanc8.f90 is highly recommended (you may find both programs on the course web page).

6 Extra credit. Write a program that calculates an integral from f(x) in the region [a, b] using Gauss quadratures for 10 points. Coefficients for Gauss quadratures can be found in Abramowitz and Stegun "Handbook of Mathematical Functions", or on the web. Calculate the integrals in the part 3, and study the efficiency and accuracy of the Gauss method for these integrals. Were there integrals that you have computed with the Gauss method, but not the Newton-Cotes method?

7 Very good extra credit. Write a program that utilizes an adaptive integration based on one of regular methods (Newton-Cotes or Gauss quadratures). Evaluate the above integrals (part 3) with your adaptive routine.

Homework 4: Random numbers. Monte Carlo integration (Due on Friday, March 26, 2010 by 21:00)

1 Study quality of a built-in random number generator (in C++ or Fortran):
Evaluate 3-rd moment of the random number distribution
Evaluate near-neighbor correlation
Plot 2D distribution for two random sequences x_i and y_i
Plot 2D distribution for correlation (x_i, x_{i+5})
How does the 3-rd moment (and/or the near-neighbor correlation) depend on the number of random numbers in your study? You may start from 4 random numbers and increase by 2 on each step.

2 Use the Monte Carlo integration (mean value method) to evaluate following integrals

a) $\int_0^{\pi/2} \cos(x)dx$

b) $\int_0^1 \frac{1}{1-0.998x^2}dx$

c) $\int_0^{2\pi} x\sin(30x)\cos(50x)dx$

d) $\int_0^1 \frac{x}{e^x-1}dx$

Study dependences of your results on number of random points.

3 Evaluate 7-D integral using the Monte Carlo integration

$$\int_0^1 dx_1 \int_0^1 dx_2 \int_0^1 dx_3 \int_0^1 dx_4 \int_0^1 dx_5 \int_0^1 dx_6 \int_0^1 dx_7 (x_1 - x_2 + x_3 - x_4 + x_5 - x_6 + x_7)^3$$

Is there a simple way to test your code for a 7-D integral?

4 Extra credit. Write a program that generates a non-uniform set of random numbers using von Neumann rejection. Apply the program for the normal distribution

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-x_0)^2}{2\sigma^2}\right)$$

Use $\sigma = 1.0$ and $x_0 = 0.0$. Plot the distribution as a histogram (you may use Excel for plotting).

5 Extra credit. Write a program that uses the Metropolis sampling for more efficient Monte Carlo integration. Apply the program to evaluate integrals 2a) and 2b), and study the efficiency of Metropolis sampling.

Homework 5: Systems of linear equations & eigenvalue problem (Due on April 21, 2010 by 13:30)

Solve either part 1 or part 2.

1 Write a program that solves a system of n linear equations using Gauss elimination (with scaling and partial pivoting). As a starting point you may use the programs Gauss_2.f90 from the course program library.

Apply the program to solve the following systems

a) $\begin{bmatrix} 1 & 3 & 2 & -1 \\ 4 & 2 & 5 & 1 \\ 3 & -3 & 2 & 4 \\ -1 & 2 & -3 & 5 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 9 \\ 27 \\ 19 \\ 14 \end{bmatrix}$ and b) $\begin{bmatrix} 2 & -2 & 2 & 1 \\ 2 & -4 & 1 & 3 \\ -1 & 3 & -4 & 2 \\ 2 & 4 & -3 & -2 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 7 \\ 10 \\ -14 \\ 1 \end{bmatrix}$

2 Write a program that computes all eigenvalues of a real symmetric matrix using Jacobi method. As a starting point you may use the program Jacobi.f90 from the course library.

Apply the program to computer eigenvalues (and eigenvectors if you can) for the following matrices

a) $\begin{bmatrix} 18 & 6 & 4 \\ 6 & 10 & 3 \\ 4 & 3 & 12 \end{bmatrix}$ and b) $\begin{bmatrix} 1 & 2 & 3 \\ 2 & 2 & -2 \\ 3 & -2 & 1 \end{bmatrix}$