

Physics 234: Computational Physics

Final Exam

Thursday, April 22, 2010 / 14:00–17:00 / CAB 239

Student's Name: _____

Instructions

There are eleven questions. You should attempt all of them. Mark your response on the test paper in the space provided. For those questions that ask you to sketch a diagram, please provide meaningful labels. You are allowed one 8.5" × 11" sheet of notes (written on both sides). No other aids are permitted.

Good luck!

<i>question</i>	<i>points awarded</i>
1	/2
2	/5
3	/4
4	/4
5	/4
6	/3
7	/4
8	/2
9	/4
10	/4
11	/4
	<hr/>
	/40

Exam questions

1. The elements of a symmetric $N \times N$ matrix A satisfy $A_{ij} = A_{ji}$ for all $i, j = 1, 2, \dots, N$. There are N diagonal elements and $N^2 - N$ off-diagonal elements, only half of which are unique (since they are equal in pairs). Hence, there are

$$N_p = N + \frac{N(N-1)}{2} = \frac{N(N+1)}{2}$$

independent quantities that must be kept in memory. A *packed* storage scheme is one that stores only the minimum N_p elements.

Consider two possible ways in which the matrix A can be flattened into one-dimensional arrays $a = (a_0 \ a_1 \ \dots \ a_{N_p-1})$ and $b = (b_0 \ b_1 \ \dots \ b_{N_p-1})$. In the first, the *upper* triangular elements are stored in column-major order:

$$A = \begin{pmatrix} A_{11} & A_{12} & A_{13} & \cdots & A_{1N} \\ A_{21} & A_{22} & A_{23} & \cdots & A_{2N} \\ \vdots & & & \ddots & \vdots \\ A_{N1} & A_{N2} & A_{N3} & \cdots & A_{NN} \end{pmatrix} = \begin{pmatrix} a_0 & a_1 & a_3 & a_6 & \cdots & a_{N_p-N} \\ & a_2 & a_4 & a_7 & & \\ & & a_5 & a_8 & & \\ & & & a_9 & & \\ & & & & \ddots & \\ & & & & & a_{N_p-1} \end{pmatrix}$$

In the second, the *lower* triangular elements are stored in column-major order:

$$A = \begin{pmatrix} A_{11} & A_{12} & A_{13} & \cdots & A_{1N} \\ A_{21} & A_{22} & A_{23} & \cdots & A_{2N} \\ \vdots & & & \ddots & \vdots \\ A_{N1} & A_{N2} & A_{N3} & \cdots & A_{NN} \end{pmatrix} = \begin{pmatrix} b_0 & & & & \\ b_1 & b_N & & & \\ b_2 & b_{N+1} & b_{2N-1} & & \\ b_3 & b_{N+2} & b_{2N} & b_{3N-3} & \\ \vdots & & & & \ddots \\ b_{N-1} & & & & b_{N_p-1} \end{pmatrix}$$

(2 points) Which of the following `return` statements to `double A(int, int)` correctly implements either of the packed storage schemes described above? Circle your answers.

```
const int N = ??, Np = N*(N+1)/2;
double a[Np], b[Np];
```

```
double A(int i, int j) { assert(i > 0 and i <= N and j > 0 and j <= N);
```

- (a) `return a[i-1+j*(j-1)/2];` (b) `return b[i+(2*N-1-j)*j/2];`
 (c) `return a[i+j*(j+1)/2];` (d) `return b[i-1+(2*N-j)*(j-1)/2];`
 (e) `return a[N*i+j];` (f) `return b[i+N*j];`
 (g) `return a[N*(i-1)+j-1];` (h) `return a[i-1+N*(j-1)];`

```
}
```

2. A fictitious 20-bit floating-point type has 1 sign bit, 7 exponent bits, and 12 fraction bits. The exponent field is interpreted with an offset of 63. In other words,

$$[s/e_6e_5 \cdots e_1e_0/f_{11}f_{10} \cdots f_1f_0] = (-1)^s \times 2^{(e_6e_5 \cdots e_1e_0)_2 - 63} \times (1 + .f_{11}f_{10} \cdots f_1f_0)_2.$$

The largest possible exponent value (all bits on) is reserved to signal error states, such as `inf` and `nan`; the smallest (all bits off) marks the representation as denormalized. As usual, the leading bit of the significand is hidden and has value 1 for normalized numbers and value 0 for denormalized numbers.

- (a) (1 point) In base 2,

$$\pi \doteq 11.001001000011111101101010100010 \cdots$$

How is this number stored in the 20-bit floating-point format described above? (Use rounding rather than truncation.)

/ /

- (b) (1 point) What are the largest and smallest positive (nonzero) values that can be represented?

/ /

/ /

- (c) (1 point) Is it possible to represent zero?

- (d) (2 points) What is the floating-point difference between π and $9/8$?

/ /

3. The arctan function can be expanded in a variety of ways—for example, as an infinite Taylor series or as an infinite continued fraction:

$$\arctan x = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \frac{x^9}{9} - \dots = \frac{x}{1 + \frac{x^2}{3 + \frac{4x^2}{5 + \frac{9x^2}{7 + \dots}}}}$$

These two expressions, systematically truncated at higher and higher order, yield the sequences

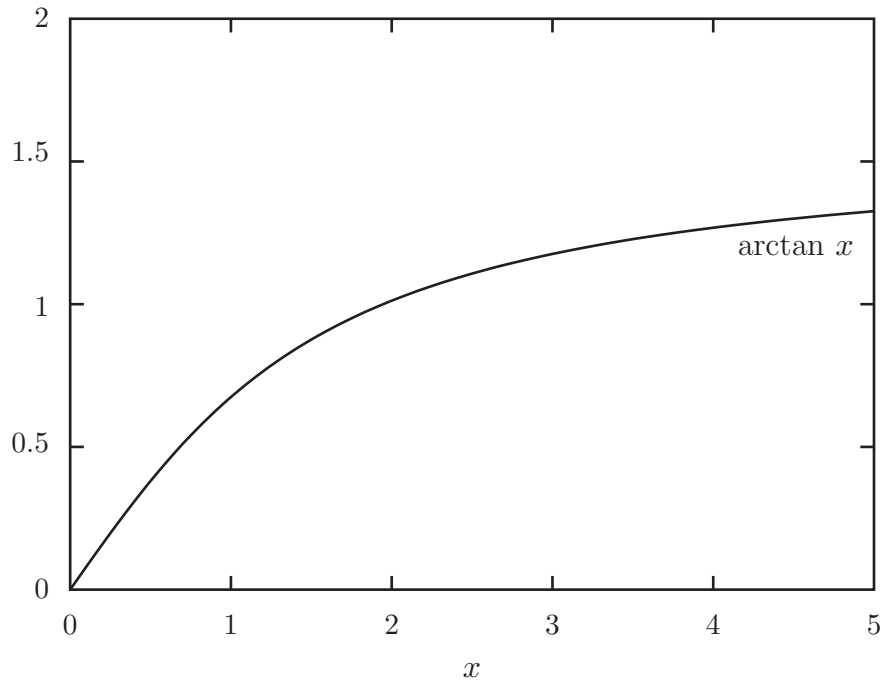
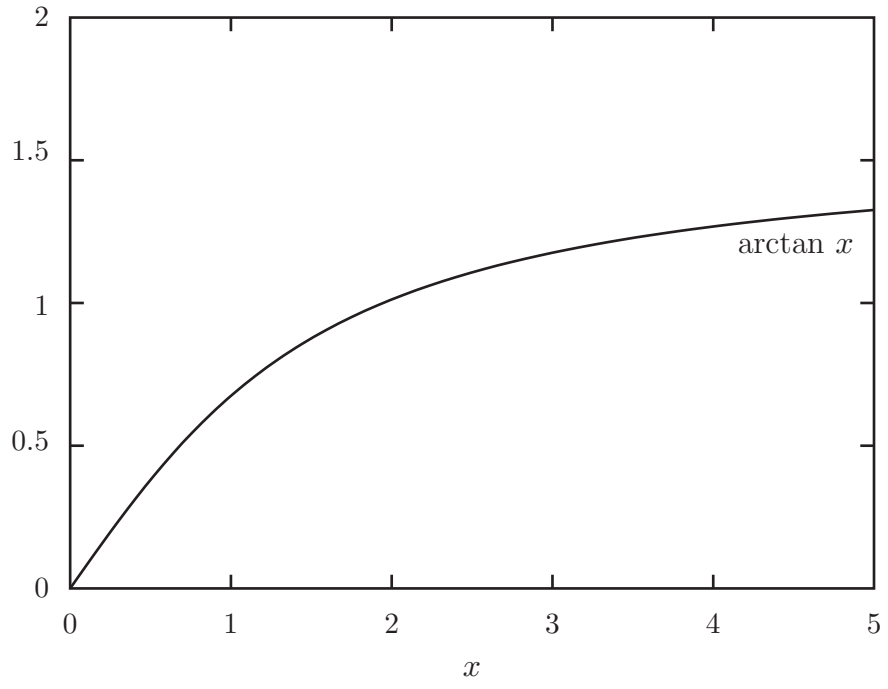
$$t_0(x) = x, \quad t_1(x) = x - \frac{x^3}{3}, \quad \dots, \quad t_n(x) = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \dots + (-1)^n \frac{x^{2n+1}}{2n+1}$$

and

$$c_0(x) = x, \quad c_1(x) = \frac{x}{1+x^2}, \quad \dots, \quad c_n(x) = \frac{x}{1 + \frac{x^2}{3 + \frac{4x^2}{5 + \dots + \frac{n^2 x^2}{2n+1}}}}$$

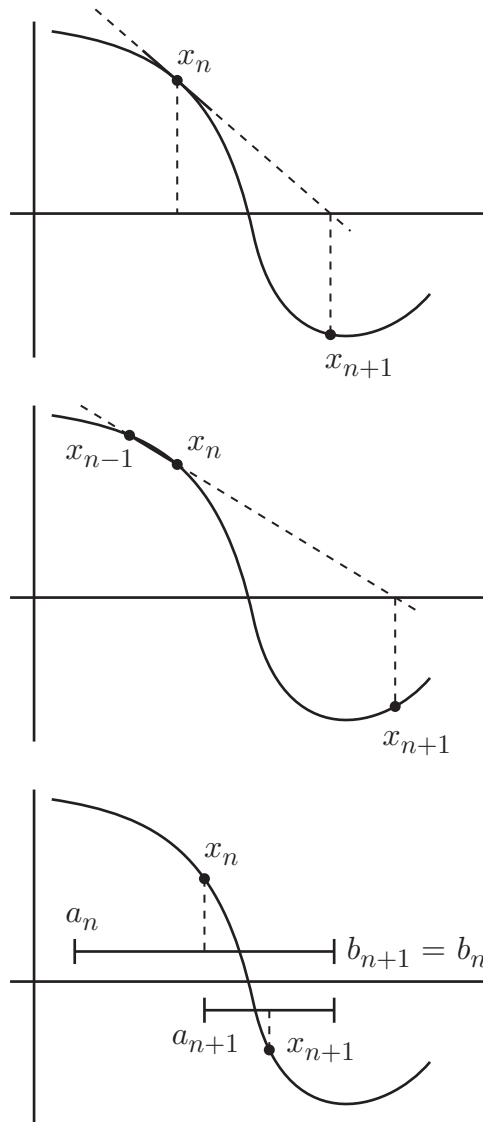
both of which converge pointwise to $\arctan x$ (in the limit $n \rightarrow \infty$).

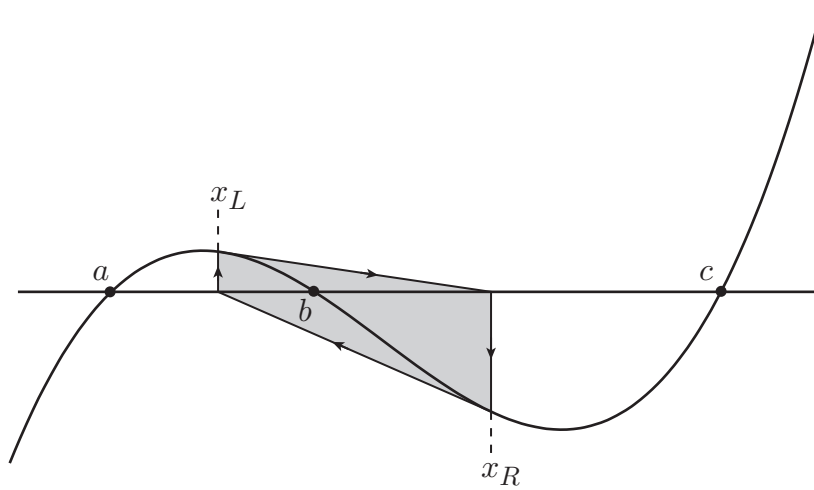
- (a) (2 points) Sketch the four curves $t_0(x), \dots, t_3(x)$ and the four curves $c_0(x), \dots, c_3(x)$ on the next page in the top and bottom panels, respectively.
- (b) (2 points) Discuss the large- x behaviour of $t_n(x)$ and $c_n(x)$. How do they differ? And why might you choose one sequence over the other for your numerical work?



4. Three different root-finding methods are shown pictorially in the figure below.

- (a) (3 points) Name each method and indicate its (asymptotic) rate of convergence (for the case of a simple root).
- (b) (1 point) Tell me which, if any of them, offers a guarantee of convergence and under what circumstances.





5. The cubic function $f(x) = (x-a)(x-b)(x-c)$ has three real roots. Newton's root-finding method applied to $f(x)$ exhibits a finite basin of attraction (x_L, x_R) around the point b . The boundary of that basin is defined by the critical cycle shown in the figure above.

(a) (1 point) Newton's equation is an iterative rule that generates an improved guess x_{n+1} from the previous guess x_n . Write down Newton's equation in terms of $f(x)$ and its first derivative.

(b) (1 point) The sequence $(x_n) = (x_0, x_1, x_2, \dots)$ converges to b whenever $x_0 \in (x_L, x_R)$. What can you say about the limit for initial guesses that fall outside this basin of attraction?

(c) (2 points) Show that x_L and x_R , the two end points of the basin of attraction, satisfy

$$(x_R - x_L) [(x_R - b)(x_R - c) + (x_R - a)(x_R - c) + (x_R - a)(x_R - b)] = (x_R - a)(x_R - b)(x_R - c)$$

and the same equation again with x_L and x_R swapped.

6. Suppose that the function

$$g(x) = \frac{x^2}{\sqrt{1+x^2} - 1}$$

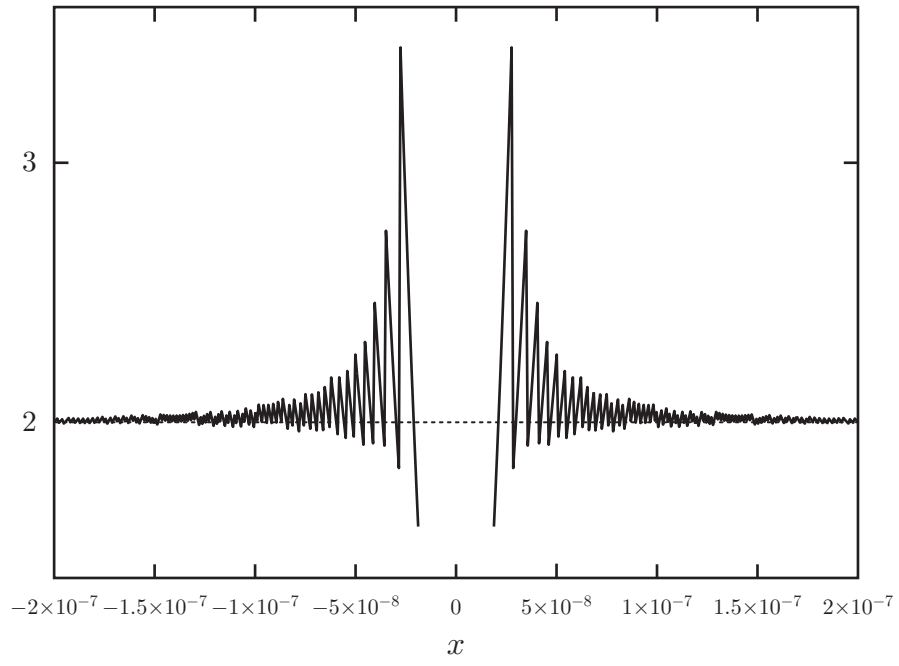
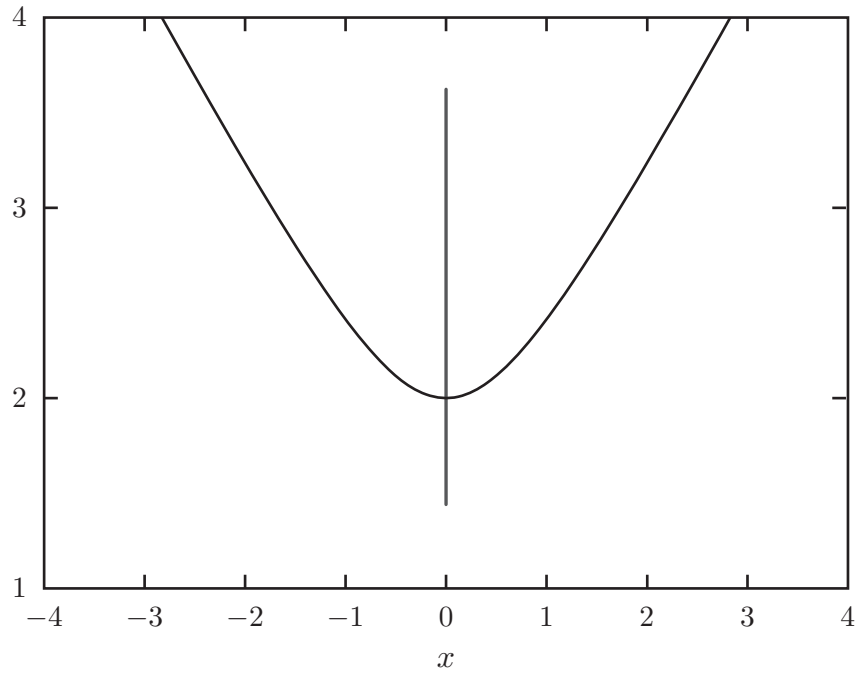
is computed (in naïve fashion) with the code

```
#include <cmath>
using std::sqrt;

double g(double x) { return x*x/( sqrt(1+x*x) - 1 ); }
```

to produce the data plotted on the next page. Note that the lower panel is simply a magnification of the upper one. For values $|x| \lesssim 2 \times 10^{-7}$, the computation as implemented fails to reproduce the correct behaviour of $g(x)$.

- (a) (1 point) Why does the computation fail so badly? What is the source of all the numerical noise?
- (b) (2 point) Produce a carefully-written version of `double g(double)` that is numerically safe in the vicinity of $x = 0$.



7. Division of the polynomial $p(x) = x^4 - 2x^3 - x^2 - 4x + 12$ by a monomial factor $(x - a)$ can be carried out via *synthetic division*. The algorithm is illustrated in tabular form below:

$$\begin{array}{r|rrrrrr}
 a & 1 & -2 & -1 & -4 & 12 \\
 & & a & a^2 - 2a & a^3 - 2a^2 - a & a^4 - 2a^3 - a^2 - 4a \\
 \hline
 & 1 & a - 2 & a^2 - 2a - 1 & a^3 - 2a^2 - a - 4 & a^4 - 2a^3 - a^2 - 4a + 12
 \end{array}$$

Note that the bottom-right entry is exactly $p(a)$, and thus it vanishes whenever a is a root of $p(x)$. Moreover, $p(a)$ represents the remainder after the polynomial division, and the coefficients of the quotient polynomial are given by the other terms on the bottom row.

For example,

$$\begin{array}{r|rrrrr}
 1 & 1 & -2 & -1 & -4 & 12 \\
 & & 1 & -1 & -2 & -6 \\
 \hline
 & 1 & -1 & -2 & -6 & \mathbf{6}
 \end{array}$$

tells us not only that $p(1) = 6$ but also that $p(x) = (x - 1)(x^3 - x^2 - 2x - 6) + 6$. Similarly,

$$\begin{array}{r|rrrrr}
 3 & 1 & -2 & -1 & -4 & 12 \\
 & & 3 & 3 & 6 & 6 \\
 \hline
 & 1 & 1 & 2 & 2 & \mathbf{18}
 \end{array}$$

implies that $p(x) = (x - 3)(x^3 + x^2 + 2x + 2) + 18$.

- (a) (2 points) Compute the synthetic division tableau for $(x - 2)$ into $p(x)$.

- (b) (1 point) Determine the largest integer m such that $p(x) = (x - 2)^m q(x)$, where $q(x)$ is a quotient polynomial of degree $4 - m \geq 0$. (In other words, how many times does $(x - 2)$ divide $p(x)$ with no remainder?)

- (c) (1 point) Show how to evaluate the polynomial $p(x)$ in a C function using at most three multiplications (*) and three additions/subtractions (+/-). You may store intermediate values and employ parentheses to modify the order of operations.

```
double p(double x)
// function that evaluates  $p(x) = x^4 - 2x^3 - x^2 - 4x + 12$ 
{

}
}
```

8. The first derivative of

$$f(x) = \frac{1 - 4x^4}{1 + 2|x|}$$

evaluated at $x_0 = -1/3$ is $f'(x_0) = 26/25 = 1.04$. Suppose that we know nothing about $f(x)$ except for its value at a few points:

$$\begin{aligned} f(-1/3) &\doteq 0.6296296296 \\ f(-13/48) &\doteq 0.6626084092 \\ f(-5/24) &\doteq 0.7112013208 \\ f(-1/12) &\doteq 0.8573082011 \\ f(1/6) &\doteq 0.7523148148 \end{aligned}$$

For convenience, we've picked $f(x_0)$ and $f(x_m) = f(x_0 + 2^{-m})$ for $m = 1, 2, 3, 4$ so that the points of evaluation are separated by decreasing powers of 2. This choice leads to four first-order finite differences of the form

$$D^{(1,m)} = \frac{f(x_0 + 2^{-m}) - f(x_0)}{2^{-m}}$$

and a simple recursion rule

$$D^{(n+1,m)} = \frac{2^n D^{(n,m+1)} - D^{(n,m)}}{2^n - 1}$$

for building the full Richardson tableau. And here it is:

$D^{(1,1)}$				0.354630			
	$D^{(2,1)}$				1.93823		
$D^{(1,2)}$		$D^{(3,1)}$		1.14643		0.602804	
	$D^{(2,2)}$		$D^{(4,1)}$		0.936660		1.10624
$D^{(1,3)}$		$D^{(3,2)}$		1.04154		1.04331	
	$D^{(2,3)}$				1.01665		
$D^{(1,4)}$				1.02910			

(a) (1 point) Sketch $f(x)$ and indicate the finite differences $D^{(1,1)}$ and $D^{(1,3)}$.

(b) (1 point) Why is the three-term extrapolation starting from $D^{(1,2)}$, $D^{(1,3)}$, $D^{(1,4)}$ and ending at $D^{(3,2)}$ considerably more accurate than the four-term extrapolation that includes $D^{(1,1)}$?

9. Numerical evaluation of the integral

$$I = \int_L^R dx f(x)$$

is performed using the following algorithm:

- i. Fill an array x with the numbers L and R and with $N - 1$ numbers drawn randomly from a uniform distribution on the interval (L, R) .
- ii. Sort x in ascending order.
- iii. Construct a cubic spline

$$s(x) = \begin{cases} s_0(x) & \text{if } L = x_0 \leq x < x_1 \\ s_1(x) & \text{if } x_1 \leq x < x_2 \\ \vdots & \\ s_i(x) & \text{if } x_i \leq x < x_{i+1} \\ \vdots & \\ s_{N-1}(x) & \text{if } x_{N-1} \leq x \leq x_N = R \end{cases}$$

with N segments passing through the $N + 1$ vertices $(x_i, f(x_i))_{i=0}^N$.

iv. Approximate the integral I by replacing the integrand $f(x)$ by $s(x)$.

(a) (2 points) Assume that each spline segment is of the form

$$s_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

and that the coefficients a_i , b_i , c_i , and d_i have been determined. Write an expression for the final approximation (step iv) in terms of the segment widths $\Delta_i = x_{i+1} - x_i$. Be sure to express your answer as a nested factorization à la Horner.

(b) (1 point) How does the accuracy of this approach scale with N ? Explain your reasoning.

(c) (1 point) Since this algorithm is partly random, its result will depend on the initial seed value of the random number generator. What distribution of values (each an independent approximation to I) might we expect if the calculation is run repeatedly for many different seeds?

10. A data set $S = \{(T_i, \rho_i)_{i=1}^N\}$ comprises N resistivity measurements ρ_i of a metallic sample at various temperatures T_i . (For simplicity, we ignore experimental error.) The theoretical prediction for the resistivity at low temperatures is

$$\rho(T) = r_0 + r_1 T^2 + r_2 \log \frac{T_K}{T} + r_3 T^5 \quad (\text{Jun Kondo, 1964}).$$

- (a) (1 point) Let's initially take the point of view that the *Kondo temperature*, T_K , is fixed and that the coefficients r_0, \dots, r_3 are the only free parameters. Argue that the algebraic conditions for a perfect fit (in which $\rho(T)$ passes through each data point) can be recast as a matrix equation $Ar = \rho$. Write the matrix A explicitly and make clear its dimensions.

(b) (2 points) Due to limitations of the theoretical modelling, it is highly unlikely that a four-parameter fit can be made to pass exactly through all the data points (numbering $N \gg 4$). So instead, we look for the best of all possible imperfect fits—viz., the one that minimizes the sum of the square of the residuals. Find an expression for the solution \hat{r} that minimizes $F(r) = \|Ar - \rho\|^2$. (Here, we use $\|v\|^2 = v^T v$ to denote the column vector norm.) I suggest that you proceed by first computing the “downhill” direction $-\nabla F(r)$ and then determining where it vanishes.

(c) (1 points) What additional complications arise if we decide to optimize the fit over T_K in addition to r_0, \dots, r_3 . Describe in detail how you might accomplish this.

11. A mass m orbits a much larger body of mass M . Let's select a two-dimensional coordinate system that coincides with the plane of the orbit: the relative position $\mathbf{r} = (x, y)$ of the satellite changes with velocity $\mathbf{v} = (v^x, v^y)$. The Newtonian dynamics under gravity can be described by two coupled first-order equations

$$\dot{\mathbf{v}} = \frac{\mathbf{F}}{\mu} = -\frac{GMm}{\mu r^2} \hat{\mathbf{r}},$$

$$\dot{\mathbf{r}} = \mathbf{v}.$$

The vector $\hat{\mathbf{r}}$ has unit magnitude and points along the line running from M to m ;

$$\mu = \left(\frac{1}{m} + \frac{1}{M} \right)^{-1}$$

is the reduced mass of the pair.

- (a) (2 points) To solve this system of equations numerically, let's discretize the time variable using a uniform time step Δt . We define $\mathbf{v}_n \equiv \mathbf{v}(t_n) \equiv \mathbf{v}(n \cdot \Delta t)$ and $\mathbf{r}_n \equiv \mathbf{r}(t_n) \equiv \mathbf{r}(n \cdot \Delta t)$. Explain in detail the approximation scheme that leads to these four Euler update equations:

$$v_{n+1}^x := v_n^x - \frac{\Delta t G(M+m)x_n}{(x_n^2 + y_n^2)^{3/2}}, \quad x_{n+1} := x_n + \Delta t v_n^x,$$

$$v_{n+1}^y := v_n^y - \frac{\Delta t G(M+m)y_n}{(x_n^2 + y_n^2)^{3/2}}, \quad y_{n+1} := y_n + \Delta t v_n^y.$$

- (b) (2 points) The total angular momentum $\mathbf{L} = \mathbf{r} \times \mu\mathbf{v}$ of the orbiting system is a conserved quantity. Show that from one time step to the next (under Euler updates) $L_n^z = \mu(x_n v_n^y - y_n v_n^x)$ is conserved only up to order $(\Delta t)^2$.