Assignment 8

1. Suppose we know $f(x)$, $f(x+h)$, $f'(x)$ and $f'(x+h)$. Find a way to combine these numbers to estimate $f''(x)$ with the highest possible order of accuracy. Said more technically, find coefficients $a(h)$, $b(h)$, $c(h)$, and $d(h)$ and an estimator

$$A(h) = af(x) + bf(x+h) + cf'(x) + df'(x+h) \ .$$

The estimator should satisfy

$$A(h) \sim f''(x) + A_1 h^{p_1} + \cdots \ .$$

with the highest possible p_1.

2. Write a routine that takes as input a, b, n, and f and uses the trapezoid rule with n equally spaced points to estimatee

$$I = \int_a^b f(x)\, dx \ .$$

Give computational evidence that the method is second order accurate for sufficiently regular functions.[1] Choose example(s) where I is known and where I is unknown. Here are some interesting examples. You should select some for results to submit, but it should be easy to try a lot of them once your code is working.

$$f(x) = \sin(x) \ , \quad a = 0, b = 1 \quad (b = \pi \text{ is a poor choice})$$
$$f(x) = \sin(40x) \ , \quad a = 0, b = 1 \quad (\text{need smaller } h \text{ to see the order of accuracy})$$
$$f(x) = \sin(x^2) \ , \quad a = 0, b = 10 \quad (\text{answer not known})$$
$$f(x) = x^r \sin(\tfrac{1}{x}) \ , \quad a = 0, b = 1, r > 0 \quad (f(0) = 0, \text{ the order of accuracy depends on } r.)$$

For the last example, find r where it's second order and r where it is not.

3. Write a routine that takes a second order accurate integrator such as the one from Exercise 2 and produces a fourth order integrator. This function should have input n, a, b, and f. It should call the second order integrator with n and $2n$ points and do one level of Richardson extrapolation. Use code similar to that of Exercise 2 to verify that your integrator is fourth order when f is smooth enough but not otherwise.

[1] This should take the form of well formatted tables with well chosen numbers of digits and output format.

4. There is a more general version of the SVD that allows more general norms on "both sides". If R is a symmetric positive definite (written SPD from now on) matrix that is $m \times m$, the $R-$norm on $\mathbb{R}^m$ is

$$\|x\|_R = \left(x^T R x \right)^{\frac{1}{2}} .$$

The R inner product is

$$\langle x, y \rangle_R = x^T R y .$$

The R norm is related to the R inner product in the same way the 2 norm is related to the ordinary inner product:

$$\|x\|_2^2 = x^T x \ , \ \ \|x\|_R^2 = \langle x, x \rangle_R .$$

Suppose V is a matrix with $p \leq m$ columns v_k:

$$V = \begin{pmatrix} \vdots & \vdots & & \vdots \\ v_1 & v_2 & \cdots & v_p \\ \vdots & \vdots & & \vdots \end{pmatrix} .$$

The vectors v_k are orthonormal in the R inner product if

$$\langle v_j, v_k \rangle_R = \delta_{jk} = \begin{cases} 1 & \text{if } j = k \\ 0 & \text{if } j \neq k \end{cases}$$

Exercise 5 uses this generalized SVD when R is a diagonal set of *weights*

$$R = \mathrm{diag}(r_1, \cdots, r_m) \iff \langle x, y \rangle_R = \sum_{j=1}^m x_j y_j r_j .$$

The matrix R is positive definite (and symmetric) if the weights are positive. The theory for the special case of "weighted ℓ^2 norms" does not seem simpler than the theory of general inner products that do not require R to be diagonal.

(a) Show that the columns of V are orthonormal in the R inner product if and only if

$$U^T R U = I_{p \times p} .$$

(b) Show that if U is square and its columns are orthonormal in the R inner product, then

$$U U^T = R^{-1} .$$

Explanation: If $R = I$, which is the "standard", ℓ^2, or "euclidian" inner product, this is simply $U^T U = I \iff UU^T = I$. The correct generalization to a general R inner product might seem harder to guess. *Hint*: One solution uses the Cholesky factorization of R, which exists only if R is symmetric and positive definite. Note that if $R = LL^T$ then $R^{-1} = L^{-T} L^{-1}$. The notation L^{-T} is possible because $\left(L^{-1}\right)^T = \left(L^T\right)^{-1}$.

(c) Let A be an $m \times n$ matrix and S a symmetric positive definite (written SPD) $m \times m$ matrix and R $n \times n$ and SPD. Consider the optimization problem

$$\max_{\|x\|_R = 1} \|Ax\|_S \ .$$

Let v_1 be an optimizer and $u_1 = \sigma_1 A v_1$ with $\|u_1\|_S = 1$. Show that

$$\langle v_1, x \rangle_R = 0 \implies \langle u_1, Ax \rangle_S = 0 \ .$$

Hint: We derived this orthogonality property in class in the special case $R = I_{n \times n}$ and $S = I_{m \times m}$. The same argument works here.

(d) Show that if v_2 is an optimizer of $\|Ax\|_S$ over vectors with $\|x\|_R = 1$ and $\langle v_1, x \rangle_R = 0$, and if $Av_2 = \sigma_2 u_2$, with $\|u_2\|_S = 1$, then v_1, v_2 are orthonormal in the R inner product and u_1, u_2 are orthonormal in the S inner product.

(e) Show that you can continue in this way to get an $m \times m$ matrix V and an $n \times n$ matrix U so that $AV = U\Sigma$ and $V^T R V = I$ and $U^T S U = I$. *Explanation*: The relation $AV = U\Sigma$ is the same relation that the ordinary SVD satisfies, though that is more often written $A = U\Sigma V^T$. The difference here is that U and V are orthogonal matrices in the S and R inner products respectively. Here, $A = U\Sigma V^T R$ (you're not asked to prove this, but it's a consequence of part (b)).

(f) Find a formula/algorithm that computes U and V, given inputs A, S, and R, using the standard SVD (available in Numpy) and the Cholesky factorizations $R = LL^T$ and $S = MM^T$. *Hint*: Compute L and M, then compute $\widetilde{V} = VL^{-T}$ and $\widetilde{U} = UM^{-T}$ using an ordinary SVD, then get V and U.

5. This exercise revisits Exercise 3 of Assignment 5. The matrices are a little modified. You have the opportunity to observe things that were missed in Assignment 5. The goal here is to *discretize* a linear problem involving functions to get finite dimensional linear algebra problems that approximate the *continuous* problem (the problem involving functions). The dimensions in the matrix approximations should be large so that the approximation is accurate, but the work and memory needed for the finite dimensional problem, the *discretized* problem, grow with the dimension.

   The continuous problem involves an *integral operator*, which is a formula of the form

$$g(x) = \int K(x, y)\, f(y)\, dy \ . \tag{1}$$

   The function K is the *integral kernel*. The operation (1) is written abstractly as $g = Kf$, where K denotes both the integral kernel function $K(x, y)$ and the action of the abstract integral operator.

The discrete (or *discretized*) approximation involves a vector of n values of u and m values of v.

$$f = \begin{pmatrix} f(y_1) \\ \vdots \\ f(y_n) \end{pmatrix} = \begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix} \quad , \qquad g = \begin{pmatrix} g(x_1) \\ \vdots \\ g(x_m) \end{pmatrix} = \begin{pmatrix} g_1 \\ \vdots \\ g_m \end{pmatrix} \quad .$$

It is common to use some notation to distinguish between the function $f(y)$ and the vector f, such as writing $f^h \in \mathbb{R}^n$ for the vector (h being a reference to a discretization length, also called *step size*). Feel free to do this if you like. The integral operator (1) is *discretized* by approximating it with an $m \times n$ matrix, also called K:

$$g = Kf \quad , \qquad g_i = \sum_j K_{ij} f_j \ . \tag{2}$$

This could be written $g^h = K^h f^h$ if you don't like using K for both the integral kernel and the matrix approximation to it. We will see that the entries of the matrix K_{ij} are not just values of the integral kernel: $K_{ij} \neq K(x_i, y_j)$.

As in Assignment 5, we take f to be a function defined on the outer r_2 ellipse, and g to be defined on the inner r_1 ellipse. Suppose p is a point on the inner ellipse, we define

$$g(p) = \int \frac{1}{|p-q|} f(q)\, dl(q) \ . \tag{3}$$

The integral is over the whole outer ellipse, q is a generic point on the outer ellipse, and dl is a unit of arc length. You can think of f as representing a light source density for light being emitted on the outer ellipse. The value $f(q)$ is the intensity at the point q. Then $g(p)$ represents the brightness at a point p on the inner ellipse.

The integral (3) can be made more explicit using θ as a parameter for the outer ellipse and writing

$$q = (s(\theta)\cos(\theta), s(\theta)\sin(\theta)) \ . \tag{4}$$

We need to take s according to (5) of Assignment 5. Then[2]

$$dl = \left| \frac{dq}{d\theta} \right| d\theta \ , \tag{5}$$

and θ can run from 0 to 2π so that q traverses the whole outer ellipse. Then we can take points $q_k = q(\theta_k)$, with $\theta_k = kh$ and $h = \frac{2\pi}{n}$. You can let k run from 0 to $n-1$ (natural in Python) or from 1 to n (natural for a mathematician). You can evaluate the derivative factor in (5) using

[2] This is a formula from vector calculus.

implicit differentiation (for $\frac{ds}{d\theta}$) and the product rule. You may find that it is easier to get the derivative formulas right if you don't write one big formula but define notation for pieces and compute and finally assemble the pieces. For example, compute (in this order)

$$\frac{ds}{d\theta}\,,\quad \frac{dq_x}{d\theta}\,,\quad \frac{dq_y}{d\theta}\,,\quad \left|\frac{dq}{d\theta}\right| = \left[\left(\frac{dq_x}{d\theta}\right)^2 + \left(\frac{dq_y}{d\theta}\right)^2\right]^{\frac{1}{2}}\,.$$

This puts the specific formula (3) in the form (1) with p as x, and θ as y, and

$$K(p,\theta) = \frac{1}{|p - q(\theta)|}\left|\frac{dq}{d\theta}\right|\,. \tag{6}$$

(a) Discretize the integral (1), (3), (6) using n equally spaced values θ_k and the trapezoid rule. Be careful to treat the beginning and end correctly, as $\theta = 0$ and $\theta = 2\pi$ refer to the same point $q = (r_2, 0)$ on the on the x axis. Take f to be a smooth function such as the one used for the movie in Assignment 5 and do a *convergence study* to determine the accuracy as a function of n. You may find the result surprising because it is not second order. Make a plot of log error as a function of n and describe what this suggests for the error as a function of n. There is no formula for the exact answer, but you can get to within machine precision, which is called the *ground truth* in accuracy studies, using a large n (but, surprisingly, not very large).

(b) Find weights r_j for the outer ellipse and s_j for the inner ellipse so that

$$\int f(q)^2\, dl(q) \approx \sum_{j=0}^{n-1} f(\theta_j)^2 r_j$$

$$\int g(p)^2\, dl(p) \approx \sum_{k=0}^{m-1} g(\theta_k)^2 s_k\,.$$

In each case, the left side should be a trapezoid rule approximation to the integral on the right using the points $q_j = q(\theta_j)$ and $p_k = p(\theta_k)$. Write code to evaluate these weights, which should use code you created for part (a).

(c) Use the algorithm of Exercise 4 to compute and plot the first few generalized singular vectors of the matrix defined in part 5a using discrete weighted inner products defined by the weights r_j and s_k in part 5b. Make plots for a enough values of m and n to see that there is a limit as n and m go to infinity.

(d) Choose m and n large enough so that the first maybe ten or twenty left and right generalized singular vectors are accurate approximations of teh $m \to \infty$ and $n \to \infty$ limit (the *continuum limit*). Make plots (one for left and another for right singular vectors) to see how the larger ones look and observe why the singular values decrease. If v_j^h is a left singular vector that approximates a continuous left singular function $v_j(q)$, why is it that

$$\sigma_j u_j(p) = \int \frac{1}{|p-q|} v_j(q)\, dl(q) \tag{7}$$

is small? This is not a question that calls for formulas, just an understanding of what happens when you plug a function v_j such as you see in your plots into an integral like (7).