1 Lecture 21

1.1 Quiz

Problem 1.

Constrained Optimization In a constrained optimization problem, when is the Hessian matrix of the Lagrangian function positive definite?

Hint: All diagonal entries of a SPD matrix must be positive.

Solution 1.1. Recall that the hessian contains a zero on a diagonal (in fact a block of zeros), so that it cannot be positive definite.

Problem 2.

Lagrange Multiplier

1 point

To minimize $f(x, y) = x^2 + 2y^2$ subject to $g(x, y) = x + 4y - 6 = 0$, we can look for critical points of the Lagrangian function $\mathcal{L}_f(x, y, \lambda) = f(x, y) + \lambda g(x, y)$. Provide the value of λ at the minimum by solving the the system of equations given by $\nabla\mathcal{L}(x, y, \lambda) = \mathbf{0}$.

Solution 1.2. Recall what the lagrangian looks like rather than combine the terms and then differentiate.

Your answer is correct.

A correct answer is: '-1.3333333333333333'.

We obtain the system of equations:

$$\mathbf{0} = \nabla\mathcal{L}(x, y, \lambda) = \begin{bmatrix} 2x + \lambda \\ 4y + 4\lambda \\ x + 4y - 6 \end{bmatrix}.$$

The first two equations imply that $x = -\lambda/2, y = -\lambda$, and inserting these in to the last, we obtain $0 = x + 4y - 6 = -\frac{9\lambda}{2} - 6$, so $\lambda = -\frac{4}{3}$.

Problem 3.

Constrained Optimization

Which of the following statements are true for finding the solution $\boldsymbol{x}^*$ of the constrained optimization problem,

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \text{ subject to } \boldsymbol{g}(\boldsymbol{x}) = \mathbf{0} \text{ and } \mathbf{h}(\mathbf{x}) \leq \mathbf{0}.$$

$\nabla f(x^*) = 0$

x^* is a critical point of the Lagrangian function.

Sequential Quadratic Programming may be unreliable unless it starts close to the solution.

The inequality constraint $h(x)$ is only active on the interior of the feasible set.

Solution 1.3. We know that a first order optimality condition is simply that $\nabla f(x^*)^T s \geq 0$, so the first answer cannot be correct(actually, the fist answer cannot be correct because KKT tells us that $\nabla_x(L, \lambda_1, \lambda_2) = 0$ and not $\nabla f(x^*) = 0$; we know that x^* is a critical point of the Lagrangian by definition; SQP may be unreliable unless it starts close to the solution, precisely because SQP is Newton, which may fail if we don't apply it close to the solution.

The last choice needs to be examined more closely.

In general, for h to be a non trivial constraint there has to be a point (and hence a neighborhood) where $h(x) > 0$ (this means that some h_i gives $h_i(x) > 0$). It follows that by continuity there is a sequence of points $\{x_n\}$ such that $h(x_n) \to 0$, meaning that $x_n \to$ some point $z \in h^{-1}(0)$. It follows that z must be some point on the boundary. UNRESOLVED

We can design a function h so taht $h(x) = 0$ and there exists a neighborhood at x called C such that there exist c_1 and c_2 such that $h(c_1) > 0$ and $h(c_2) < 0$.

Just take $h : \mathbb{R} \to \mathbb{R}$ where $h(x) = x$ and consider $x = 0$.

Problem 4.

Inequality Constrained Minimum
Find the optimal parameters and λ inside the feasible set for the minimization function $f(x, y) = 5x^2 + 3y$ and $x + y \geq 5$.

Solution 1.4. Form the lagrange function, which is

$$5x^2 + 3y + \lambda(-x - y + 5)$$

Then solve for x, y, λ such that $\nabla_L = \mathbf{0}$.

After you solve for the parameters, tweak them so that $x + y \geq 5$.

Note that, in general, one would apply Newton's method (or a similar optimization scheme) to try and determine the minimum of ∇_L.

Problem 5.

Sequential Quadratic Programming Which of the following are true about sequential quadratic programming (SQP)?

Select all that apply: A single iteration of conjugate gradient suffices for each step of SQP Each iteration of SQP requires a single iteration of Newton's method SQP is more effective at handling inequality constraints than equality constraints When dealing only with equality constraints, SQP is Newton's method for optimization

Solution 1.5.

- Conjugate gradient is not used in SQP – Newton's method is.

- Yes, this is true – by definition.
- yes, for Heath explains that when we use inequality constraints, at a given iteration, we consider those constraints that are active and include them in the Lagrangian to which we then apply Newton's method.
  - For reasons that are not explained, this complication of having varying lagrangians that we minimize is less effective than simply using one, non-varying Lagrangian throughout the course of the problem.
- Yes, indeed – when we don't have inequality constraints, we are simply evertime computing $H_L(x)s = -\nabla_L(x)$ for s

Definition 1.6. An interpolatio problem asks us to find a function f such that given a set of points $\{(x_i, y_i)\}$ it holds that $f(x_i) = y_i$ for all i.

Definition 1.7. Suppose that we define $f = \sum_{i=1}^{n} \phi_j(x)\alpha_j$. Then$f(x_i) = y_i = \sum_{k=1}^{n} \phi_k(x_i)\alpha_k$.

This can be framed as solve for α in

$$Y = V\alpha$$

where $V_{i,k} = \phi_k(x_i)$ and Y has all the solutions.

If we require V to be square, and hence there to be precisely as many basis functions as there are points n – then the system above admits a unique solution in the event that V is full rank.

Definition 1.8. When we use a monomial basis we ask to solve for $(x_1 \dots x_n)$ such that

$$p(t) = \sum_{j=0}^{n-1} x_{j+1} t^j = y_{j+1}$$

where $((t_1, y_1) \dots (t_n, y_n))$ are the function points.

To solve for the x, we can solve the following system

$$\begin{bmatrix} 1 & t_1 & t_1^2 \dots (t_1)^{n-1} \\ 1 & t_2 & t_2^2 \dots (t_2)^{n-1} \\ \vdots & & \\ 1 & t_n & t_n^2 \dots (t_n)^{n-1} \end{bmatrix} t = y$$

Suppose that $Vz = 0$ where z is some vector. This implies that a polynomial of degree $n-1$ has n roots (ie $t_1 \dots t_n$), which can only take place if the resulting polynomial is the zero polynomial, which implies that $z = 0$. This tells us that under this basis, there necessarily exists an interpolant.

Definition 1.9. Lagrange interpolation seeks to make a basis V that is the identity matrix. To that end we define

$$\phi_j = \frac{\prod_{i \neq j}(x - x_i}{\prod_{i \neq j}(x_i - x_j)}$$

Then observe that $\phi_j(x_j) = 1$ and that $\phi_j(x_i) = 0$ for any $i \neq j$.

Definition 1.10. Newton polynomials are obtained by using the basis function

$$\phi_j(x) = \prod_{k=1}^{j-1}(x - x_k)$$

where $\phi_1 = 1$. As a consequence, this gives us a matrix that is lower triangular.

Finding the α vector then is an $O(n^2)$ process.

Definition 1.11. Polynomials are orthogona iff over an interval (for example $[-1, 1]$), given some weight function $w(x)$, we have

$$\int_{-1}^{1} f(x)g(x)w(x) = 0$$

This can be shown to induce an inner product and, hence, we can invoke Gram Schmidt on a collection of n polynomials to obtain a basis for $\mathbb{P}_{n-1}$, the vector space of all polynomials of degree $n - 1$.

Remark 1.12. This is useful, because if our interpolants constitute a basis, then we can be assured that we are representing as many functions as possible; it also turns out to be true that when we solve the normal equations $Va = y$ where V is the vandermonde matrix, the matrix V^TV is diagonal (UNRESOLVED: there is some matrix that – according to Heath – ends up being diagonal, but it is not known whether this is V^TV). Orthogonal polynomials also satisfy a three term recurrence, making computation of successive polynomials tractable.

Definition 1.13. In one special case, if we let the basis functions be

$$\phi_j(x) = \cos(j \cos^{-1}(x))$$

and we use equispaced points as our nodes, ie we use

$$x_i = \cos\left(\frac{i}{k}\pi\right)$$

then the entries of the vandermonde matrix, say $V_{i,j}$ end up being

$$\cos\left(j(i/k)\pi\right)$$

It can be shown that the mat-vec of the resultant matrix by any other vector is an $n \log n$ process. This is somehow related to the fourier transform. There are other useful properties of the chebyshev polynomials that Heath discusses.

2 Lecture 23

2.1 Quiz

Problem 2.

see the code below that is commented and hence won't appear on th erendered page

Problem 3.

Let hte interpolant be $a_0 + a_1 x$.

$$\implies a_0 + a_1(41.08) = 1.5945575554$$

$$\implies a_0 + a_1(41.09) = 1.5944984844$$

Solve for a_0 and a_1
will give us an ouput that is less than $f(41.08)$.

Suppose that we estimate a function using linear interpolation. The error is then bounded by:

$$\frac{f''(\zeta)}{2!}h^n$$

Nah use the bound $h^n CM$ and take C and M to be 1 and h is by definition the distance between endpoints of the interval, which in this case is 0.04.

Problem 5.

Assume that the error bound is given by

$$h^n CM$$

<++>

Remark 2.1. We jsut learned that the error of any interpolant which is a polynomial to a function f where p is at most of degree $n-1$ satisfies the result that the error $p_{n-1}(x) - f(x)$

$$= \frac{f^n(\zeta)}{n!}\prod_{i=1}^{n}(x - x_i)$$

for some $\zeta \in I$, which is a closed intervable on which f is n times continuously differentiable.

We can use this to obtain a generic error bound on our interpolant:
Assume that $x_i \in [x_1, x_n] = h$. Then

$$\max|p_{n-1}(x) - f(x)| \leq h^n \frac{f^n(\zeta)}{n!} = h^n \frac{MC}{1}$$

where we assume that $f^n(x) \leq M$ for all $x \in I$ and C is some constant independent of h.

Remark 2.2. How to perform piecewise polynomial interpolation of degree n.

- Assume that we're given two points (x_1, y_1) and (x_2, y_2) for one interval and that there are N intervals. Then form the interpolant:

  $$a_0 + a_1 x + a_2 x^2 + ... a_n x^n = f(x)$$

  and solve for a_i using $f(x_1) = y_1$ and $f(x_2) = y_2$.

  - In general, observe that there will be $(n+1)(N)$ unknowns.
    * For cubic splines, there are $4N$ unknowns
  - This gives us two equations for one interval.
    * In total, there fore we have $2N$ equations here.

- Add some smoothness conditions – ie the derivative of the polynomial in one interval must agree with the derivative in the next interval and we get more equations. Make this hold with respect to second derivatives as well.
  - This gives us $2(N-1)$ equations.
    * Finally require that the second derivatives be zero at each of the endpoints.
      · This gives us 2 equations.
  - At this point, we have $4N$ equations. If we're working with cubic splines, then we have as many unknowns as equations and we can proceed to solve the system.
    * This construction above is called a cubic spline