Penalty Methods – General Idea

- Idea: instead of constraining the problem, add a term to the objective function which makes the function value worse as the constraint is more violated
- So convert

$$\begin{array}{lll} \min & f(\bar{x}) & \\ \text{s.t.} & h_i(\bar{x}) = 0 & \Rightarrow \quad \min \quad P(f, g_j, h_i, r) \\ & g_j(\bar{x}) \le 0 & \end{array}$$

where P is a penalty function and r is a positive penalty parameter

CPE 778, Lect. 19

1

Another Idea – the L_1 Penalty Function

- So the original penalty function gives an ill-conditioned matrix as soon as *r* gets large enough to help solve the constrained NLP. Can we use a different function?
- The L_1 penalty function replaces the *r* with finite weights, and avoids squaring the constraint:

$$\min \quad P_1(\bar{x}, \overline{w1}, \overline{w2}) = f(\bar{x}) + \left[\sum_{i=1}^{M1} w1_i \left| h_i(\bar{x}) \right| + \sum_{j=1}^{M2} w2_j \left| g_i(\bar{x}) \right| \right]$$

CPE 778, Lect. 19

2

The L_1 Penalty Function

- Advantages:
  - Works for large but finite values of the weighting parameters
- Disadvantages:
  - Discontinuous at $h_i(x)=0$, so taking derivatives may not be possible at that important point
  - What type of solutions are obtained in the intermediate steps leading to the optimum?

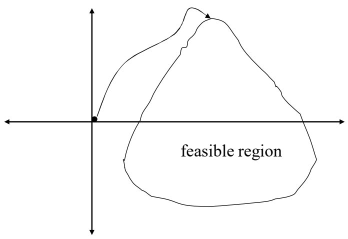
  Infeasible ones!

CPE 778, Lect. 19

3

Infeasible Intermediate Solutions

• Why do we care about the feasibility of intermediate points?



• If we never do find the optimum of a large NLP, we might choose to run at a nonoptimal, feasible point. This cannot be done if intermediate solutions are infeasible

CPE 778, Lect. 19

4

Barrier Methods – General Idea

- Remember we are trying to solve a constrained NLP by converting it to an unconstrained problem
- Idea: subtract a term from the objective function which goes to -∞ as we get near the constraint boundary
- Again, convert

$$\begin{array}{ll} \min & f(\bar{x}) \\ \text{s.t.} & h_i(\bar{x}) = 0 \\ & g_j(\bar{x}) \geq 0 \end{array} \quad \Rightarrow \quad \min \quad B(\bar{x}, r)$$

where B is a barrier function and r is a positive barrier parameter

CPE 778, Lect. 19

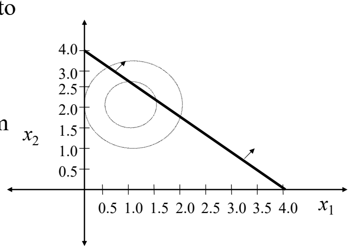
5

Example

$$\begin{array}{ll} \min & f(\bar{x}) = (x_1 - 1)^2 + (x_2 - 2)^2 \\ \text{s.t.} & g(\bar{x}) = x_1 + x_2 - 4 \geq 0 \end{array}$$

• Note the solution to the unconstrained problem, [1,2], is infeasible to the constrained problem

• Thus the constraint must be active at optimality



CPE 778, Lect. 19

6

Example

$$\min \quad f(\bar{x}) = (x_1 - 1)^2 + (x_2 - 2)^2$$
$$\text{s.t.} \quad g(\bar{x}) = x_1 + x_2 - 4 \geq 0$$

- We can write a logrithmic barrier function as

$$\min \quad B(\bar{x}, r) = f(\bar{x}) - r\ln(g(\bar{x}))$$

or for our example

$$\min \quad B(\bar{x}, r) = (x_1 - 1)^2 + (x_2 - 2)^2 - r\ln(x_1 + x_2 - 4)$$

- What are the limiting cases here?

If $(x_1 + x_2 - 4) < 0$, B is undefined (infeasible)

If $(x_1 + x_2 - 4) \to 0$, $B \to +\infty$

CPE 778, Lect. 19

7

Example, continued

$$\min \quad B(\bar{x}, r) = (x_1 - 1)^2 + (x_2 - 2)^2 - r\ln(x_1 + x_2 - 4)$$

- For a given positive r, how does the solution to this problem relate to the solution of the original problem?
- We get a feasible, nonoptimal solution to the original problem

As $r \to 0$, min $B \to$ min of constrained problem

- For very small r, we get very close to the constrained optimum <u>on the feasible side</u>

CPE 778, Lect. 19

8

Problems with Barrier Functions

- First problem: ln is discontinuous/undefined in areas of interest
- Second problem:

$$\text{As } r \to 0, \ \text{CN}(\nabla_{\bar{x}}^2 B(\bar{x}, r)) \to \infty,$$

causing much roundoff error

- So this method isn't used directly either, but these functions are used within many other algorithms

CPE 778, Lect. 19

9

Successive Linear Programming: A method which really works!

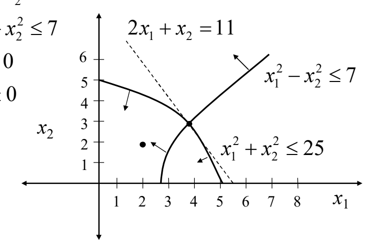
- A different idea: relax the constrained NLP to an LP, not an unconstrained NLP
- Use Taylor series to linearize everything
- Derive a series of improved linear approximations to converge on the NLP solution

CPE 778, Lect. 20

10

Another Example

$$
\begin{aligned}
\text{max} \quad & f(\bar{x}) = 2x_1 + x_2 \\
\text{s.t.} \quad & g_1(\bar{x}) = x_1^2 + x_2^2 \le 25 \\
& g_2(\bar{x}) = x_1^2 - x_2^2 \le 7 \\
& g_3(\bar{x}) = x_1 \ge 0 \\
& g_4(\bar{x}) = x_2 \ge 0
\end{aligned}
$$



- Note the optimum lies at [4,3], on the contour $f(x)$=11
- Also, [2,2] is a feasible starting point

CPE 778, Lect. 20

11

Performing the Linearizations

- SLP requires an initial feasible point. We'll start with [2,2]
- We must linearize <u>all</u> constraints and the objective function, so we allow no discontinuous or nondifferentiable functions in the problem

- Write the problem in terms of the change in each variable $\Delta\bar{x}$
- The general formula for linearizing any function is: $p(\bar{x}) = p(\bar{x}^{(0)}) + \nabla p(\bar{x}^{(0)})^T (\Delta\bar{x})$

CPE 778, Lect. 20

12

Setting up an LP subproblem

- OK, now write a Taylor series approximation to each function, around the point [2,2]

$$\begin{aligned} \max \quad & f(\bar{x}) = 2x_1 + x_2 \\ \text{s.t.} \quad & g_1(\bar{x}) = x_1^2 + x_2^2 \le 25 \\ & g_2(\bar{x}) = x_1^2 - x_2^2 \le 7 \\ & g_3(\bar{x}) = x_1 \ge 0 \\ & g_4(\bar{x}) = x_2 \ge 0 \end{aligned}$$

$$p(\bar{x}) = p(\bar{x}^{(0)}) + \nabla p(\bar{x}^{(0)})^T (\Delta \bar{x})$$

CPE 778, Lect. 20

13

Setting up an LP subproblem

- Here's what I get:

$$\begin{aligned} \max \quad & f(\bar{x}) \approx 2x_1^{(0)} + x_2^{(0)} + [2,1]\begin{bmatrix}\Delta x_1 \\ \Delta x_2\end{bmatrix} = 2\Delta x_1 + \Delta x_2 + 6 \\ \text{s.t.} \quad & g_1(\bar{x}) = (x_1^{(0)})^2 + (x_2^{(0)})^2 + [2x_1, 2x_2]\begin{bmatrix}\Delta x_1 \\ \Delta x_2\end{bmatrix} \le 25 \Rightarrow 4\Delta x_1 + 4\Delta x_2 + 8 \le 25 \\ & g_2(\bar{x}) = (x_1^{(0)})^2 - (x_2^{(0)})^2 + [2x_1, -2x_2]\begin{bmatrix}\Delta x_1 \\ \Delta x_2\end{bmatrix} \le 7 \Rightarrow 4\Delta x_1 - 4\Delta x_2 \le 7 \\ & g_3(\bar{x}) = x_1^{(0)} + [1,0]\begin{bmatrix}\Delta x_1 \\ \Delta x_2\end{bmatrix} \ge 0 \Rightarrow 2 + \Delta x_1 \ge 0 \\ & g_4(\bar{x}) = x_2^{(0)} + [0,1]\begin{bmatrix}\Delta x_1 \\ \Delta x_2\end{bmatrix} \ge 0 \Rightarrow 2 + \Delta x_2 \ge 0 \end{aligned}$$

CPE 778, Lect. 20

14

Setting up an LP subproblem

- We also need bounds on the Δx's, to keep the Taylor series approximations reasonably accurate:

$$-1 \le \Delta x_1 \le 1, \ -1 \le \Delta x_2 \le 1$$

- Now put all these together and simplify:

$$\begin{aligned} \max \quad & f(\bar{x}) = 2\Delta x_1 + \Delta x_2 \\ \text{s.t.} \quad & g_1(\bar{x}) = \Delta x_1 + \Delta x_2 \le 4.25 \\ & g_2(\bar{x}) = \Delta x_1 - \Delta x_2 \le 1.75 \\ & -1 \le \Delta x_1 \le 1 \\ & -1 \le \Delta x_2 \le 1 \end{aligned}$$

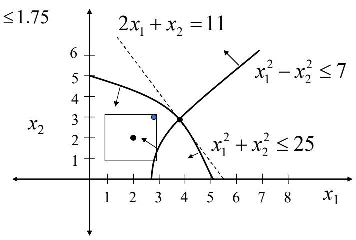
- This is called an LP subproblem. Solve using the simplex method

CPE 778, Lect. 20

15

LP subproblem #1

$$\begin{aligned}
\max \quad & f(\bar{x}) = 2\Delta x_1 + \Delta x_2 \\
\text{s.t.} \quad & g_1(\bar{x}) = \Delta x_1 + \Delta x_2 \le 4.25 \\
& g_2(\bar{x}) = \Delta x_1 - \Delta x_2 \le 1.75 \\
& -1 \le \Delta x_1 \le 1 \\
& -1 \le \Delta x_2 \le 1
\end{aligned}$$



- Note the optimum lies at [3,3] – we moved as far as we could. No original constraint is active.

CPE 778, Lect. 20

16

LP subproblem #2

- Now, check and make sure [3,3] is better than [2,2], to make sure we're converging.
- $f(2,2)=6$, $f(3,3)=9$, OK
- Now set up the next LP, by linearizing around [3,3]:

$$\begin{aligned}
\max \quad & f(\bar{x}) = 2\Delta x_1 + \Delta x_2 \\
\text{s.t.} \quad & g_1(\bar{x}) = \Delta x_1 + \Delta x_2 \le 7/6 \\
& g_2(\bar{x}) = \Delta x_1 - \Delta x_2 \le 7/6 \\
& -1 \le \Delta x_1 \le 1 \\
& -1 \le \Delta x_2 \le 1
\end{aligned}$$

CPE 778, Lect. 20

17

Further Iterations

- The solution to LP #2 is [4,3.167]
- Note this is infeasible to the original NLP – not a problem, but don't use intermediate solutions!
- After the third LP, we get to [4,3.005], still infeasible but very close to the optimum
- This method does not always converge, but an improvement does:
- Penalty Successive Linear Programming changes the step bounds in each step using a penalty function, causing the method to never step too far and thus always converge
- This method is used in many commercial NLP codes, but sometimes converges slowly

CPE 778, Lect. 20

18

Conclusions

- Penalty function and barrier function methods do not work well to solve NLPs as stand-alone algorithms, but many other methods use such functions to improve convergence
- Successive Linear Programming converts the NLP to a series of LPs. This does work well, if we change the step bounds using a penalty function. Convergence can be slow for many-variable problems

CPE 778, Lect. 20

19

GRG: The Generalized Reduced Gradient Method

- Iteratively solves a constrained NLP without relaxation
- Includes a fast unconstrained NLP solver, and deals with each constraint separately
- Implemented in CONOPT within the GAMS modeling system

CPE 778, Lect. 21

20

GRG: Ideas

- If we could put all constraints into the objective function, or start from a feasible point at which no constraint is active, then we could perform at least one step of an unconstrained NLP algorithm and improve our initial guess. How do we make that step?

1. Compute grad *f(x)* at the current point
2. Test to see if the iterations are converging
3. Compute a search direction based on grad *f(x)* (like steepest descent) plus other information
4. Perform a line search to figure out how far to move in the search direction
5. Repeat if still "unconstrained"

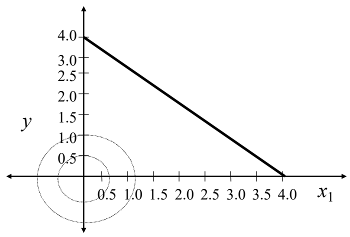
CPE 778, Lect. 21

21

GRG: Dealing with Constraints

- Simplest case – a linear equality constraint
  - We can solve for one variable and eliminate it from the other constraints and from the objective function
- Example:

$$\begin{array}{ll} \min & x^2 + y^2 \\ \text{s.t.} & x + y = 4 \end{array}$$



CPE 778, Lect. 21

22

Basic Variables for GRG

$$\begin{array}{ll} \min & x^2 + y^2 \\ \text{s.t.} & x + y = 4 \end{array}$$

- Rearrange the constraint to get $x = 4 - y$
- Call the variable being substituted out of the problem <u>basic</u>
- All other variables are considered <u>non-basic</u>

CPE 778, Lect. 21

23

Substituting In

- Put this in the objective function:

$$F(y) = (4 - y)^2 + y^2$$

- Now we can solve by setting the derivative =0 (GRG does this automatically for differentiable functions):

$$\nabla F(y) = -2(4 - y) + 2y = 0$$

- This is called the <u>reduced gradient</u>, hence the name GRG

CPE 778, Lect. 21

24

Using Steepest Descent

- Solve this to find *x,y*:

$$-8+4y=0 \Rightarrow y=2$$
$$\text{from constraint, } x=2$$

- What if we used the descent algorithm here? We could get a new value for *y* using

$$y_{i+1} = y_i + \alpha\, d$$

- Start with a feasible point – how do we find one?
  Note we have one degree of freedom in the constraints – set *y*=0, solve for *x*: [4,0]

CPE 778, Lect. 21

25

More Steepest Descent

- Now we get

$$y_{i+1} = 0 + \alpha\, d$$

- What is d?
  The negative gradient:

$$d = -\nabla F(y_0)$$
$$= -(-8+4y) \text{ at } y=0 \Rightarrow d=8$$
$$\text{so } y_1 = 0 + 8\alpha$$

CPE 778, Lect. 21

26

Performing the Line Search

- Now we need to find α: Solve

$$\min\ F(y_1)$$

- This becomes:

$$\min F(8\alpha) = (4-8\alpha)^2 + (8\alpha)^2$$

- Solve by taking the derivative again:

$$\frac{df}{d\alpha} = 2(4-8\alpha)\cdot -8 + 16\alpha = 0$$
$$= -64 + 256\alpha \Rightarrow \alpha = \tfrac{1}{4}$$

So $y_1 = 0 + 8\cdot 0.25 = 2, x_1 = 2$ (optimal solution)

CPE 778, Lect. 21

27

GRG: Details

- What if the equality constraint is nonlinear?

GRG uses an approximation procedure similar to Newton's method to solve for the basic variables

- How do we handle inequalities?

Add slack variables

CPE 778, Lect. 21

28

Superbasic Variables

- If we have inequality bounds on variables, these can cause convergence of GRG to be slow if we wish to remain feasible at every iteration
- If we allow some variables to violate their bounds, those variables are called <u>superbasic</u> for a given iteration
- GRG within GAMS is efficient because it makes wise choices for basic, non-basic and superbasic variables

CPE 778, Lect. 21

29

Basic Algorithm

- Tabu search is a stochastic algorithm which is primarily used to solve large NLP and MINLP problems. As with simulated annealing, we need a way to escape local minima (particularly if they are shallow).
- To set up the algorithm, you need to define a <u>move</u>, that is, a way to change the variables of the system to get from one feasible solution to the next. In Tabu search, the size (distance) of the move can change based on the local environment, and based on the <u>history</u> of the search process

CPE 778, Lect. 22

30

Tabu Search: Basic Ideas

- As with all stochastic algorithms, TS requires an initial feasible solution. It can be parallelized naively just by running multiple simultaneous searches, and comparing results at the end
- At each iteration, TS computes a set of random moves, and accepts the best move unless the move makes the solution infeasible, or the new point is on the Tabu list.
  - A prespecified number of previously visited solutions are stored in the Tabu list
  - This forces the search out of shallow minima, and prevents cycling
  - A very good solution can be revisited by overruling the Tabu restriction.

CPE 778, Lect. 22

31

More Uses of the Tabu List

- After a few moves, we can tell if the solutions are improving or not. If we look to be in a local well, then use local intensification: decrease the size of the moves to narrow in on a likely good local minimum
- If the area is unpromising, then use diversification: make a large move in a random direction, and restart the search. Do not reset the Tabu list, in case the new area has been searched in the recent past
- Using the history of the search in this way is called an Adaptive Memory Strategy

CPE 778, Lect. 22

32

Adaptive Memory

- The use of information from previous iterations is common in numerical analysis, not just in optimization. Can you think of any mathematical methods which use previous history to help lead to an improved solution?
  - In control theory, one uses the integral of the process output over time to help predict how the controlled variable will change in the future, and thus design a corrective action
  - In conjugate gradient optimization, both current and prior derivative information is used to derive the next iterate. This leads to better search directions than Newton's method, which only uses current information

CPE 778, Lect. 22

33

Adjustable Parameters

- TS has a number of adjustable parameters, which should be varied in order to improve solutions times on a given problem. They include:
  - Length of the Tabu list
  - Number of solutions considered in each step
  - Number of iterations with no improvement prior to diversification
  - Amount of improvement required to initiate local intensification
  - For problems with continuous variables, how "close" two solutions must be for them to be considered identical

CPE 778, Lect. 22

34

Integer Programs in Engineering

- Many engineering problems require the use of discrete variables, to represent
  - Numbers of batches
  - Fixed choices of equipment types
  - Locations of equipment, facilities, wells
  - Tasks to be performed within a given time period
  - Network connections: piping, electrical grids, roads, heat exchangers, reactors, molecules
  - Selection of solvents, catalysts, reaction chemistries
- Often, assuming continuous variables and rounding leads to a (significantly) nonoptimal solution

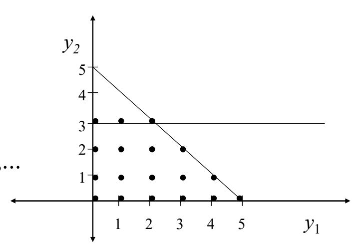
CPE 778, Lect. 23

35

Feasible Region: MILP

- Sketch the feasbile region for

$$\begin{aligned} \text{max} \quad & 2y_1 + y_2 \\ \text{s.t.} \quad & y_1 + y_2 \le 5 \\ & y_2 \le 3 \\ & y_1, y_2 = 0,1,2,... \end{aligned}$$



- Note this is an MILP, and the solution is at [5,0] – a corner of the simplex

CPE 778, Lect. 23

36

Classification of Integer Programs

- Specific types of integer programs have been named and some have specialized solvers:

- All variables binary ([0,1]): binary program, BIP
- All constraints and objective linear, both continuous and integer variables: MILP
  - Like an LP, only global solutions
  - Must deal with combinatorial explosion
- All variables integer, all equations linear: pure IP
- Anything with nonlinear functions & some integer or binary variables: MINLP
  - Clearly the toughest, many local solutions, combinatorial explosion

CPE 778, Lect. 23

37

Example: Chemical Engineering

- A consultant has n jobs which have been offered to her. Each task requires w_i hours to perform, and the consultant will be paid v_i dollars if she completes the i^{th} task. Select a subset of the tasks which can be performed within W total hours which maximizes the consultant's income. (Called a knapsack problem)
- Formulate this and compute the number of possible (not just feasible) solutions.

$$\max \quad f(\bar{y}) = \sum_{i=1}^{n} v_i y_i$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_i y_i \le W, \quad y_i = 0,1$$

$$\text{number of possible solutions}: 2^n$$

CPE 778, Lect. 23

38

Example: Economics

- A sales rep must visit clients in n cities. The distance between each city is known, and the sales rep needs to visit each client once, and return home. What order should the cities be visited in to minimize the distance traveled? (Called the Traveling Salesman Problem or TSP)

$$\min \quad f(\bar{y}) = \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} y_{ij}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} y_{ij} = 1, \ \sum_{j=1}^{n} y_{ij} = 1, \quad y_{ij} = 0,1 \ , \ y_{ii} = 0$$

- Is that all we need?

CPE 778, Lect. 23

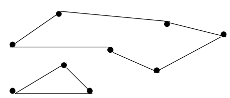
39

Assignment problems and TSP's

$$\min \quad f(\bar{y}) = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij} y_{ij}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} y_{ij} = 1, \ \sum_{j=1}^{n} y_{ij} = 1, \quad y_{ij} = 0,1 \ , \ y_{ii} = 0$$

- The problem as stated is called an assignment problem, and allows solutions like:



- To avoid these solutions, we need a "no subtours" constraint – adds many equations and variables

40

TSP's in Chemical Engineering

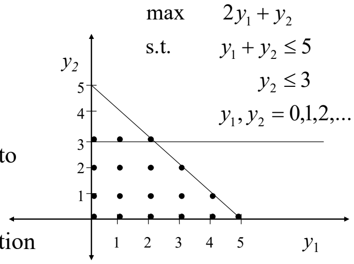
- Given a batch reactor in a flexible chemical plant which can make n specialty chemicals, derive the order in which you make each product. You know the changeover costs to switch from making product i to product j, you must make all n chemicals, and you wish to minimize total production costs
- This is a TSP as well – very tough to solve for large n

41

Bounding the Solution

- What happens if we remove the integrality restrictions?
- We get an LP, the solution to which is always ≥ the solution to the MILP (an <u>upper bound</u>)
- We can use the solution at any feasible point as a <u>lower bound</u>

$$\max \quad 2y_1 + y_2$$
$$\text{s.t.} \quad y_1 + y_2 \le 5$$
$$y_2 \le 3$$
$$y_1, y_2 = 0,1,2,\ldots$$



- The branch-and-bound method brings these bounds together to narrow the search until the optimum is found

42

Conclusions

- Problem Formulation ideas:
    - Binary/integer variables are heavily used in industrial engineering applications, when rounding leads to huge errors
    - "Selection" problems are often modeled with binary variables, for which special relationships exist that can be used to avoid nonlinearities
    - Remember to only use an integer/binary variable when necessary, since each one increases the complexity of the problem greatly

CPE 778, Lect. 23

43