

# Optimization

L. T. Biegler  
Chemical Engineering Department  
Carnegie Mellon University  
Pittsburgh, PA 15213  
biegler@cmu.edu  
<http://dynopt.cheme.cmu.edu>

March 16, 2000

Introduction

Unconstrained Optimization

Newton and quasi-Newton Methods

Constrained Optimization

Kuhn-Tucker Conditions

Successive Quadratic Programming

Reduced Gradient Methods

References and Software

# Introduction

Optimization: given a system or process, find the best solution to this process within constraints.

Objective Function: indicator of "goodness" of solution, e.g., cost, yield, profit, etc.

Decision Variables: variables that influence process behavior and can be adjusted for optimization.

In many cases, this task is done by trial and error (through case study). Here, we are interested in a *systematic* approach to this task - and to make this task as efficient as possible.

Some related areas:

- Math programming
- Operations Research

Currently - Over 30 journals devoted to optimization with roughly 200 papers/month - a fast moving field!

# Viewpoints

Mathematician - characterization of theoretical properties of optimization, convergence, existence, local convergence rates.

Numerical Analyst - implementation of optimization method for efficient and "practical" use. Concerned with ease of computations, numerical stability, performance.

Engineer - applies optimization method to real problems. Concerned with reliability, robustness, efficiency, diagnosis, and recovery from failure.

# Optimization Literature

## - Engineering

1. Edgar, T.F., and D.M. Himmelblau, Optimization of Chemical Processes. McGraw-Hill, 1988.
2. Papalambros, P.Y. and D.J. Wilde, Principles of Optimal Design. Cambridge Press, 1988.
3. Reklaitis, G.V., A. Ravindran, and K. Ragsdell, Engineering Optimization. Wiley, 1983.
4. Biegler, L. T., I. E. Grossmann and A. Westerberg, Systematic Methods of Chemical Process Design, Prentice Hall, 1997.

## - Numerical Analysis

1. Dennis, J.E. and R. Schnabel, Numerical Methods of Unconstrained Optimization, Prentice-Hall, 1983.
2. Fletcher, R. Practical Methods of Optimization, Wiley, 1987.
3. Gill, P.E, W. Murray and M. Wright, Practical Optimization, Academic Press, 1981.
4. Nocedal, J. and S. Wright, Numerical Optimization, Springer, 1998

# UNCONSTRAINED MULTIVARIABLE OPTIMIZATION

Problem:            Min         $f(x)$   
(n variables)

Note:    Equivalent to    Max  $-f(x)$   
    $x \in \mathbb{R}^n$

## **Organization of Methods**

### Nonsmooth Function

- Direct Search Methods
- Statistical/Random Methods

### Smooth Functions

- 1st Order Methods
- *Newton Type Methods*
- Conjugate Gradients

## Example: Optimal Vessel Dimensions

What is the optimal L/D ratio for a cylindrical vessel?

### Constrained Problem

$$\begin{aligned} & \text{Min } \left\{ C_T \frac{\Pi D^2}{2} + C_S \Pi D L = \text{cost} \right\} \\ (1) \quad & \text{s.t. } V - \frac{\Pi D^2 L}{4} = 0 \\ & D, L \geq 0 \end{aligned}$$

### Convert to Unconstrained (Eliminate L)

$$\begin{aligned} & \text{Min } \left\{ C_T \frac{\Pi D^2}{2} + C_S \frac{4V}{D} = \text{cost} \right\} \\ (2) \quad & \frac{d(\text{cost})}{dD} = C_T \Pi D - \frac{4VC_S}{D^2} = 0 \end{aligned}$$

$$\begin{aligned} D &= \left( \frac{4V}{\Pi} \frac{C_S}{C_T} \right)^{1/3} & L &= \left( \frac{4V}{\Pi} \right)^{1/3} \left( \frac{C_T}{C_S} \right)^{2/3} \\ \implies L/D &= C_T/C_S \end{aligned}$$

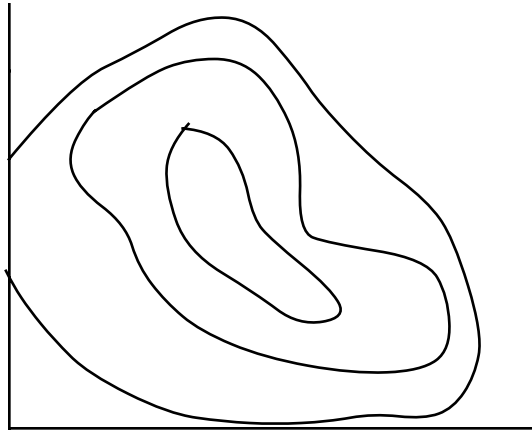
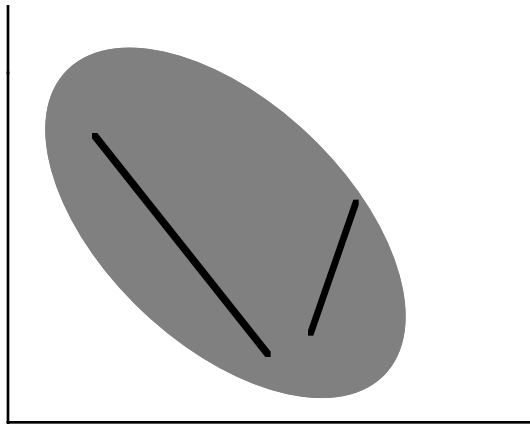
Note:

- What if L cannot be eliminated in (1) explicitly? (strange shape)
- What if D cannot be extracted from (2)? (cost correlation implicit)

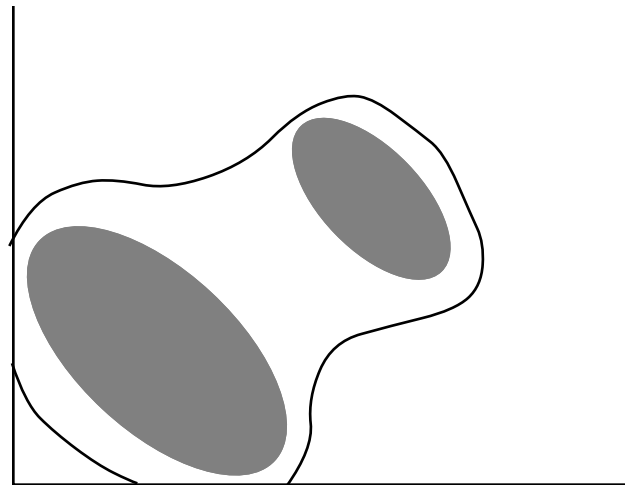
# Two Dimensional Contours of $f(\mathbf{X})$

Convex Function

Nonconvex Function

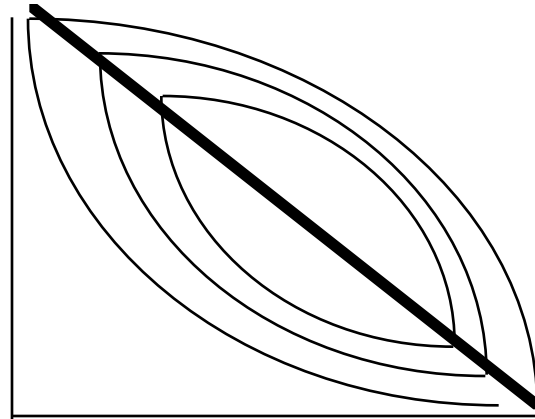
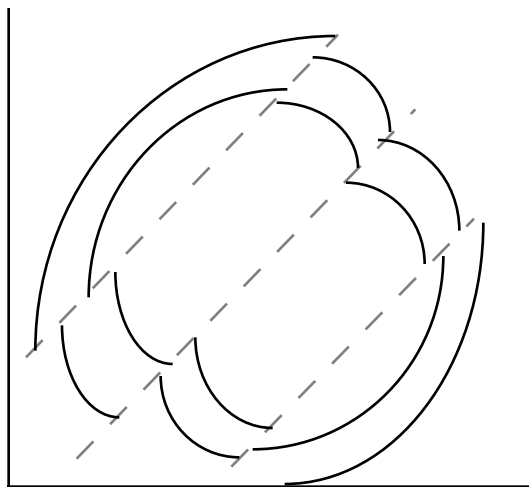


Multimodal, Nonconvex



Discontinuous

Nondifferentiable (convex)



## Some Important Points

We will develop methods that will find a *local minimum* point  $x^*$  for  $f(x)$  for a feasible region defined by the constraint functions:  $f(x^*) \leq f(x)$  for all  $x$  satisfying the constraints in some neighborhood around  $x^*$

Finding and verifying *global solutions* to this NLP will not be considered here. It requires an exhaustive (spatial branch and bound) and it much more expensive.

A local solution to the NLP is also a global solution under the following *sufficient* conditions based on convexity.

A convex function  $\phi(x)$  for  $x$  in some domain  $X$ , if and only if it satisfies the relation:

$$\phi(\alpha \xi + (1-\alpha) \eta) \leq \alpha \phi(\xi) + (1-\alpha) \phi(\eta)$$

for any  $\alpha$ ,  $0 \leq \alpha \leq 1$ , at all points in  $\xi$  and  $\eta$  in  $X$ .

# UNCONSTRAINED METHODS THAT USE DERIVATIVES

Gradient Vector - ( $\nabla f(\mathbf{x})$ )

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \dots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

Hessian Matrix ( $\nabla^2 f(\mathbf{x})$  - Symmetric)

$$\nabla^2 f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \dots & \dots & \dots & \dots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Note:  $\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial^2 f}{\partial x_j \partial x_i}$

## Eigenvalues (Characteristic Values)

Find  $x$  and  $\lambda$  where  $Ax_i = \lambda_i x_i, i = 1, n$

Note:  $Ax - \lambda x = (A - \lambda I) x = 0$  or  $\det(A - \lambda I) = 0$

For this relation  $\lambda$  is an eigenvalue and  $x$  is an eigenvector of  $A$ .

If  $A$  is symmetric, all  $\lambda_i$  are real

$\lambda_i > 0, i = 1, n$ ;  $A$  is positive definite

$\lambda_i < 0, i = 1, n$ ;  $A$  is negative definite

$\lambda_i = 0, \text{ some } i$ :  $A$  is singular

Quadratic Form can be expressed in Canonical Form (Eigenvalue/Eigenvector)

$$x^T A x \Rightarrow A X = X \Lambda$$

$X$  - eigenvector matrix ( $n \times n$ )

$\Lambda$  - eigenvalue (diagonal) matrix =  $\text{diag}(\lambda_i)$

If  $A$  is symmetric, all  $\lambda_i$  are real and  $X$  can be chosen orthonormal ( $X^{-1} = X^T$ ). Thus,  $A = X \Lambda X^{-1} = X \Lambda X^T$

For Quadratic Function:

$$Q(x) = a^T x = 1/2 x^T A x$$

define:

$$z = x^T X \quad \text{and} \quad Q(x) = Q(z) = x = Xz$$

$$a^T X z + 1/2 z^T \Lambda z$$

Minimum occurs at (if  $\lambda_i > 0$ )

$$x = -A^{-1}a \quad \text{or} \quad x = Xz = -X(\Lambda^{-1} X^T a)$$

## Example

$$\text{Min} \quad \begin{bmatrix} 1 \\ 1 \end{bmatrix}^T \mathbf{x} + \frac{1}{2} \mathbf{x}^T \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \mathbf{x} = Q(\mathbf{x})$$

$$\mathbf{A} \mathbf{X} = \mathbf{X} \mathbf{\Lambda} \quad \mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

$$\mathbf{\Lambda} = \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix} \quad \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

- All eigenvalues positive
- Minimum at  $\mathbf{z}^* = -\mathbf{\Lambda}^{-1} \mathbf{X}^T \mathbf{a}$

$$\mathbf{z}_1 = \frac{1}{\sqrt{2}} (x_1 - x_2) \quad x_1 = \frac{1}{\sqrt{2}} (z_1 + z_2)$$

$$\mathbf{z}_2 = \frac{1}{\sqrt{2}} (x_1 + x_2) \quad x_2 = \frac{1}{\sqrt{2}} (-z_1 + z_2)$$

$$\mathbf{z} = \mathbf{X}^T \mathbf{x} \quad \mathbf{x} = \mathbf{X} \mathbf{z}$$

$$\mathbf{z}^* = \begin{bmatrix} 0 \\ 2/3\sqrt{2} \end{bmatrix} \quad \mathbf{x}^* = \begin{bmatrix} 1/3 \\ 1/3 \end{bmatrix}$$

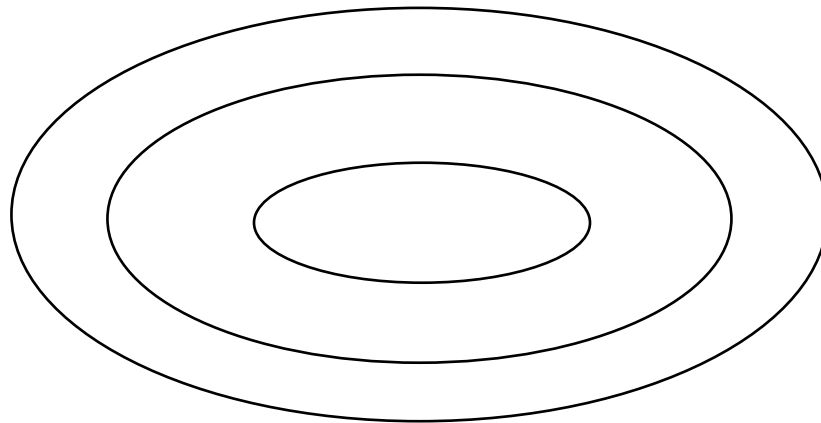
Optimality Conditions: what characterizes a (locally) optimal solution?

For smooth functions, why are contours around optimum elliptical?

Taylor Series in n dimensions about  $x^*$

$$f(x) = f(x^*) + \nabla f(x^*)^T (x - x^*) + \frac{1}{2} (x - x^*)^T \nabla^2 f(x^*) (x - x^*) + \dots$$

Since  $\nabla f(x^*) = 0$ ,  $f(x)$  is purely quadratic for  $x$  close to  $x^*$



## NEWTON'S METHOD

### Taylor Series for $f(x)$ about $x^i$

Take derivative wrt  $x$ , set LHS  $\approx 0$

$$0 \approx \nabla f(x) = \nabla f(x^i) + \nabla^2 f(x^i) (x - x^i)$$

$$\Rightarrow (x - x^i) \equiv d = - (\nabla^2 f(x^i))^{-1} \nabla f(x^i)$$

Notes:

1.  $f(x)$  is convex (concave) if  $\nabla^2 f(x)$  is all positive (negative) semidefinite  
i.e.  $\lambda_i \geq 0$  ( $\lambda_i \leq 0$ )
2. Method can fail if:
  - $x^0$  far from optimum
  - $\nabla^2 f$  is singular at any point
  - function is not smooth
3. Search direction,  $d$ , requires solution of linear equations.
4. Near solution  $\|x^{k+1} - x^*\| = K \|x^k - x^*\|^2$

## BASIC NEWTON ALGORITHM

0. Guess  $x^0$ , Evaluate  $f(x^0)$ ,  $g(x^0)$  and  $h(x^0)$ .
1. At  $x^i$ , evaluate  $\nabla f(x^i)$ ,  $\nabla g(x^i)$ ,  $\nabla h(x^i)$ .
2. Evaluate  $B^i$  from Hessian or an approximation.
3. Solve:  $B^i d = -\nabla f(x^i)$

If convergence error is less than tolerance:

e.g.,  $\|\nabla f(x^i)\| \leq \epsilon$  and  $\|d\| \leq \epsilon$  STOP, else go to 4.

4. Find  $\alpha$  so that  $0 < \alpha \leq 1$  and  $f(x^i + \alpha d) < f(x^i)$  sufficiently (Each trial requires evaluation of  $f(x)$ )
5.  $x^{i+1} = x^i + \alpha d$ . Set  $i = i + 1$  Go to 1.

## Notes on Algorithm

1. Choice of  $B^i$  determines method.

e.g.  $B^i = I/\|\nabla f(x^i)\|$                       Steepest Descent

$B^i = \nabla^2 f(x)$                                       Newton

2. With suitable  $B^i$ , performance may be good enough if  $f(x^i + \alpha d)$  merely improves (instead of minimized).

3. Local rate of convergence (local speed) depends on choice of  $B^i$ .

- Newton - Quadratic Rate

$$\lim_{k \rightarrow \infty} \frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|^2} = K$$

- Steepest Descent - Linear Rate

$$\lim_{k \rightarrow \infty} \frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|} = < 1$$

- Desired - Superlinear Rate

$$\lim_{k \rightarrow \infty} \frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|} = 0$$

4. Trust region extensions to Newton's method provide for very strong global convergence properties and very reliable algorithms.

## QUASI-NEWTON METHODS

### Motivation:

- Need  $B^i$  to be positive definite.
- Avoid calculation of  $\nabla^2 f$ .
- Avoid solution of linear system for  $d = - (B^i)^{-1} \nabla f(x^i)$

Strategy: Define matrix updating formulas that give  $(B^i)$  symmetric, positive definite and satisfy:

$$(B^{i+1})(x^{i+1} - x^i) = (\nabla f^{i+1} - \nabla f^i) \quad (\text{Secant relation})$$

DFP Formula (Davidon, Fletcher, Powell, 1958, 1964)

$$B^i = B^i + \frac{(y - B^i s)y^T + y(y - B^i s)^T}{y^T s} - \frac{(y - B^i s)^T s y y^T}{(y^T s)(y^T s)}$$

or for  $H^i = (B^i)^{-1}$

$$H^{i+1} = H^i + \frac{ss^T}{s^T y} - \frac{H^i y y^T H^i}{y H^i y}$$

where:  $s = x^{i+1} - x^i$   
 $y = \nabla f(x^{i+1}) - \nabla f(x^i)$

BFGS Formula (Broyden, Fletcher, Goldfarb, Shanno)

$$B^{i+1} = B^i + \frac{yy^T}{y^T s} - \frac{B^i s s^T B^i}{s^T B^i s}$$

or for  $H^i = (B^i)^{-1}$

$$H^{i+1} = H^i + \frac{(s - H^i y) s^T + s (s - H^i y)^T}{y^T s} - \frac{(s - H^i y) s^T y s s^T}{(y^T s)(y^T s)}$$

Notes:

- 1) Both formulas are derived under similar assumptions and have symmetry
- 2) Both have superlinear convergence and terminate in  $n$  steps on quadratic functions. They are identical if  $\alpha$  is minimized.
- 3) BFGS is more stable and performs better than DFP, in general.
- 4) For  $n \leq 100$ , these are the best methods for general purpose problems.

## CONSTRAINED OPTIMIZATION (Nonlinear Programming)

Problem:      Min  $f(x)$   
                     $x$   
                    s.t.  $g(x) \leq 0$   
                     $h(x) = 0$

where:

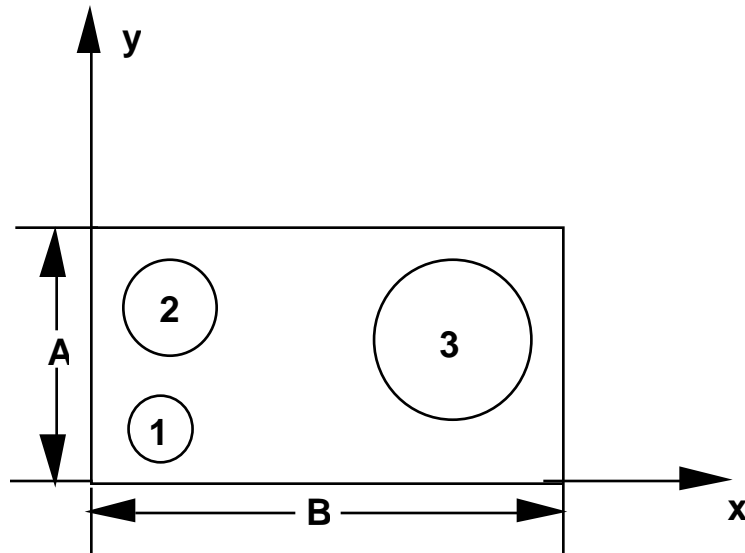
$f(x)$  - scalar objective function  
 $x$  -  $n$  vector of variables  
 $g(x)$  - inequality constraints,  $m$  vector  
 $h(x)$  - meq equality constraints.  
degrees of freedom =  $n$  - meq.

### Sufficient Condition for Unique Optimum

- $f(x)$  must be *convex*, and
- feasible region must be convex,  
i.e.  $g(x)$  are all *convex*  
 $h(x)$  are all *linear*

Except in special cases, there is no guarantee that a local optimum is global if sufficient conditions are violated.

Example: Minimize Packing Dimensions



What is the smallest box for three round objects?

Variables:  $A, B, (x_1, y_1), (x_2, y_2), (x_3, y_3)$

Fixed Parameters:  $R_1, R_2, R_3$

Objective: Minimize Perimeter =  $2(A+B)$

Constraints: Circles remain in box, can't overlap

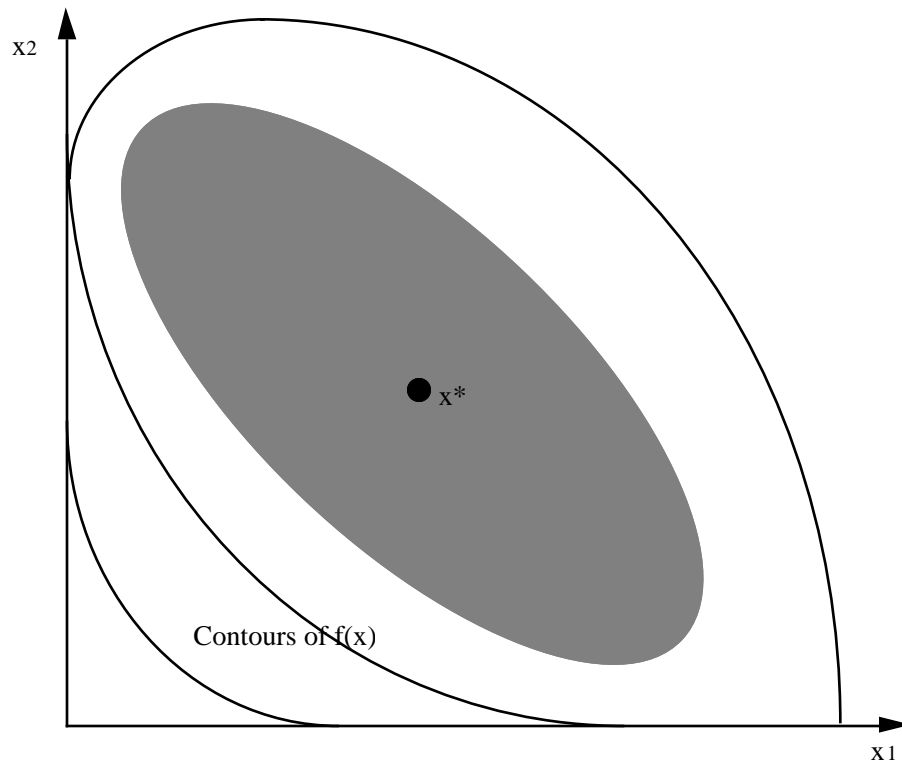
Decisions: Sides of box, centers of circles.

$$\text{in box} \begin{cases} x_1, y_1 \geq R_1 & x_1 \leq B - R_1, y_1 \leq A - R_1 \\ x_2, y_2 \geq R_2 & x_2 \leq B - R_2, y_2 \leq A - R_2 \\ x_3, y_3 \geq R_3 & x_3 \leq B - R_3, y_3 \leq A - R_3 \end{cases}$$

$$\text{no overlaps} \begin{cases} (x_1 - x_2)^2 + (y_1 - y_2)^2 \geq (R_1 + R_2)^2 \\ (x_1 - x_3)^2 + (y_1 - y_3)^2 \geq (R_1 + R_3)^2 \\ (x_2 - x_3)^2 + (y_2 - y_3)^2 \geq (R_2 + R_3)^2 \end{cases}$$

$$x_1, x_2, x_3, y_1, y_2, y_3, A, B \geq 0$$

# WHAT ARE THE CONDITIONS THAT CHARACTERIZE AN OPTIMAL SOLUTION?

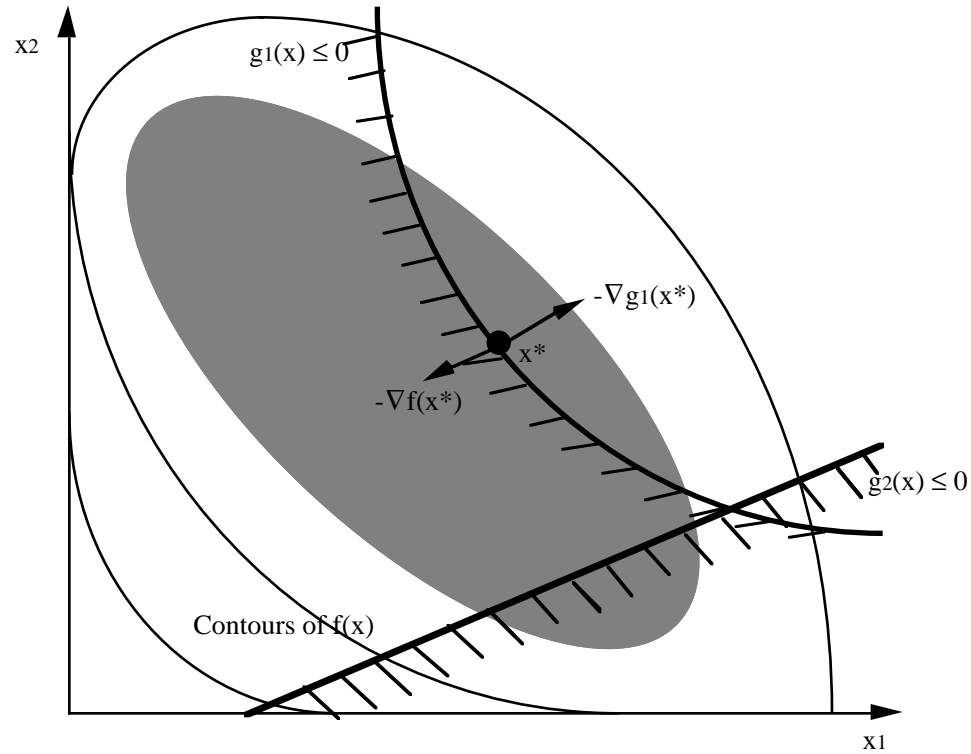


## Unconstrained Problem - Local Min.

$$\begin{aligned}\nabla f(x^*) &= 0 \\ y^T \nabla^2 f(x^*) y &\geq 0 \quad \text{for all } y \\ &\text{(pos. def.)}\end{aligned}$$

## Necessary Conditions

# OPTIMAL SOLUTION FOR INEQUALITY CONSTRAINED PROBLEM

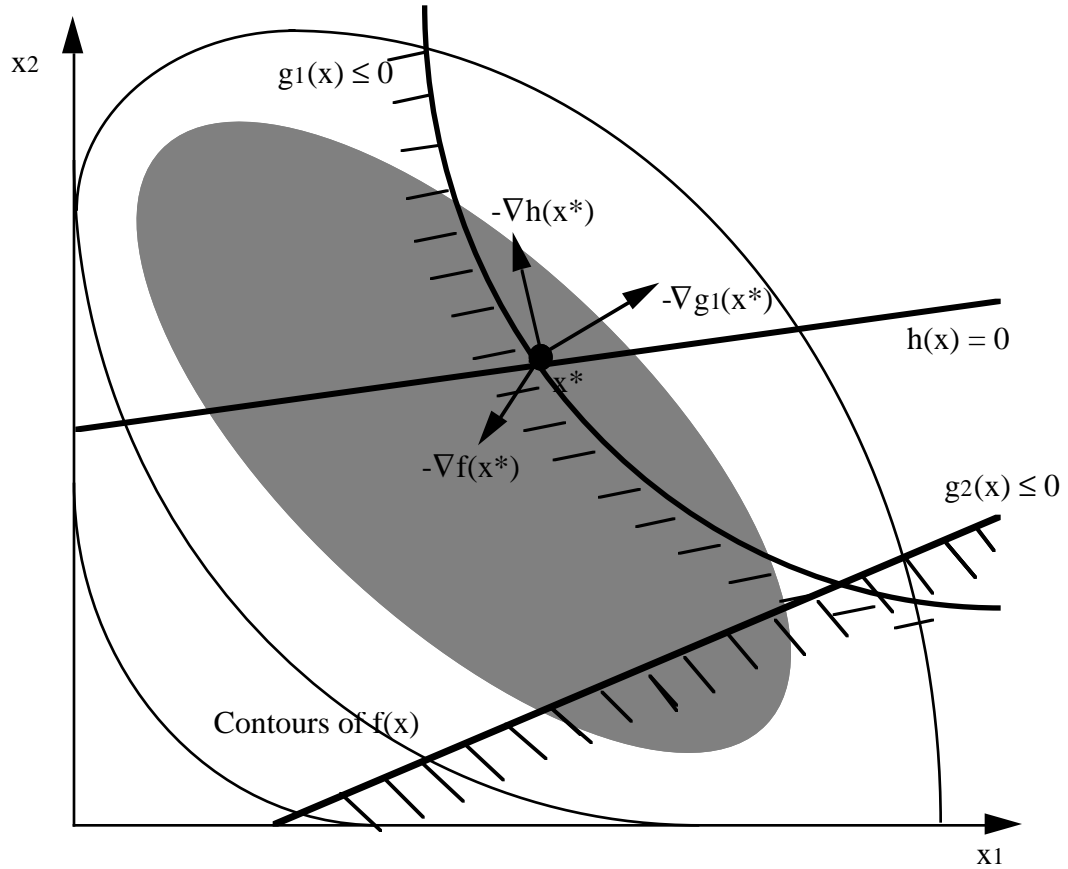


$$\begin{array}{ll} \text{Min} & f(x) \\ \text{s.t} & g(x) \leq 0 \end{array}$$

Analogy: Ball rolling down valley pinned by fence

Note: Balance of forces ( $\nabla f$ ,  $\nabla g_1$ )

# OPTIMAL SOLUTION FOR GENERAL CONSTRAINED PROBLEM



Problem:  $\text{Min } f(x)$   
s.t.  $g(x) \leq 0$   
 $h(x) = 0$

Analogy: Ball rolling on rail pinned by fences

Balance of forces:  $\nabla f, \nabla g_1, \nabla h$

# OPTIMALITY CONDITIONS FOR LOCAL OPTIMUM

## 1st Order Kuhn - Tucker Conditions (Necessary)

$$\nabla L(x^*, u, v) = \nabla f(x^*) + \nabla g(x^*)u + \nabla h(x^*) = 0$$

(Balance of Forces)

$$u \geq 0$$

(Inequalities act in 1 direction)

$$g(x^*) \leq 0, \quad h(x^*) = 0$$

(Feasibility)

$$u^T g(x^*) = 0$$

(Complementarity - either  $u_i$  or  $g_i = 0$  at constraint or off)

- $u, v$  are "weights" for "forces," known as
- Kuhn - Tucker & Lagrange multipliers
  - Shadow prices
  - Dual variables

## 2nd Order Conditions (Sufficient)

- Positive curvature in "constraint" directions.
- $p^T \nabla^2 L(x^*) p > 0$   
where  $p$  are all constrained directions.

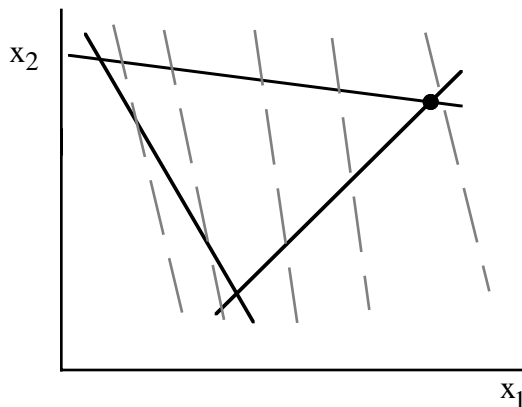
# SPECIAL "CASES" OF NONLINEAR PROGRAMMING

Linear Programming:

$$\begin{array}{ll} \text{Min} & c^T x \\ \text{s.t.} & Ax \leq b \\ & Cx = d, \quad x \geq 0 \end{array}$$

Functions are all *convex*  $\Rightarrow$  global min.

Because of Linearity, can prove solution will always lie at vertex of feasible region.



## Simplex Method

- Start at vertex
- Move to adjacent vertex that offers most improvement
- Continue until no further improvement

Notes:

- 1) LP has wide uses in planning, blending and scheduling
- 2) Canned programs widely available.

Example: Simplex Method

$$\begin{array}{ll} \text{Min} & -2x_1 - 3x_2 \\ \text{s.t.} & 2x_1 + x_2 \leq 5 \\ & x_1, x_2 \geq 0 \end{array} \Rightarrow \begin{array}{ll} \text{Min} & -2x_1 - 3x_2 \\ \text{s.t.} & 2x_1 + x_2 + x_3 = 5 \\ & x_1, x_2, x_3 \geq 0 \\ & \text{(add slack variable)} \end{array}$$

Now, define  $f = -2x_1 - 3x_2$

$$\Rightarrow f + 2x_1 + 3x_2 = 0$$

Set  $x_1, x_2 = 0, x_3 = 5$  and form tableau

$x_1$	$x_2$	$x_3$	$f$	$b$	$x_1, x_2$
					nonbasic
2	1	1	0	5	$x_3$
					basic
2	3	0	1	0	

To decrease  $f$ , increase  $x_2$ . How much? so  $x_3 \geq 0$

$x_1$	$x_2$	$x_3$	$f$	$b$
2	1	1	0	5
-4	0	-3	1	-15

$f$  can no longer be decreased! Optimal

The underlined terms are -(reduced gradients) for nonbasic variables ( $x_1, x_3$ ).  $x_2$  is basic at optimum.

## QUADRATIC PROGRAMMING

Problem:  $\text{Min } a^T x + 1/2 x^T b x$   
 $A x \leq b$   
 $C x = d$

- 1) Can be solved using LP-like techniques:  
(Wolfe, 1959)

$$\begin{aligned} \Rightarrow \text{Min } & \sum (z^+ + z^-) \\ \text{s.t. } & a + Bx + A^T u + C^T v = z^+ - z^- \\ & Ax - b + x = 0 \\ & Cx - d = 0 \\ & z^+, z^- \geq 0 \\ & \{u_i s_i = 0\} \end{aligned}$$

with complicating conditions.

- 2) If B is positive definite, QP solution is unique.  
If B is pos. semidef., optimum value is unique.
- 3) Other methods for solving QP's (faster)
- Complementary Pivoting (Lemke)
  - Range, Null Space methods (Gill, Murray).

## PORTFOLIO PLANNING PROBLEM

Formulation using LP's and QP's

### Definitions:

$x_i$  - fraction or amount invested in security  $i$

$r_i(t)$  - (1 + rate of return) for investment  $i$  in year  $t$ .

$\mu_i$  - average  $r(t)$  over  $T$  years, i.e.

$$\mu_i = \frac{1}{T} \sum_{t=1}^T r_i(t)$$

### LP Formulation

$$\begin{array}{ll} \text{Max} & \sum \mu_i x_i \\ & \sum x_i \leq C \quad (\text{total capital}) \\ \text{s.t.} & x_i \geq 0 \\ & \text{etc.} \end{array}$$

Note: we are only going for maximum average return and there is no accounting for risk.

Definition of Risk - fluctuation of  $r_i(t)$  over investment (or past) time period.

To minimize risk, minimize variance about portfolio mean (risk averse).

Variance/Covariance Matrix, S

$$\{S\}_{ij} = \sigma_{ij}^2 = \frac{1}{T} \sum_{t=1}^T (r_i(t) - \mu_i)(r_j(t) - \mu_j)$$

Quadratic Program

$$\begin{aligned} \text{Min} \quad & x^T S x \\ & \sum x_j = 1 \\ \text{s.t.} \quad & \sum \mu_j x_j \geq R \\ & x_j \geq 0, \dots \end{aligned}$$

Example: 3 investments

		$\mu_j$
1.	IBM	1.3
2.	GM	1.2
3.	Gold	1.08

$$S = \begin{bmatrix} 3 & 1 & -0.5 \\ 1 & 2 & 0.4 \\ -0.5 & 0.4 & 1 \end{bmatrix}$$

GAMS 2.25 PC AT/XT

SIMPLE PORTFOLIO INVESTMENT PROBLEM (MARKOWITZ)

```
4
5  OPTION LIMROW=0;
6  OPTION LIMXOL=0;
7
8  VARIABLES IBM, GM, GOLD, OBJQP, OBJLP;
9
10 EQUATIONS E1,E2,QP,LP;
11
12 LP.. OBJLP =E= 1.3*IBM + 1.2*GM + 1.08*GOLD;
13
14 QP.. OBJQP =E= 3*IBM**2 + 2*IBM*GM - IBM*GOLD
15 + 2*GM**2 - 0.8*GM*GOLD + GOLD**2;
16
17 E1..1.3*IBM + 1.2*GM + 1.08*GOLD =G= 1.15;
18
19 E2.. IBM + GM + GOLD =E= 1;
20
21 IBM.LO = 0.;
22 IBM.UP = 0.75;
23 GM.LO = 0.;
24 GM.UP = 0.75;
25 GOLD.LO = 0.;
26 GOLD.UP = 0.75;
27
28 MODEL PORTQP/QP,E1,E2/;
29
30 MODEL PORTLP/LP,E2/;
31
32 SOLVE PORTLP USING LP MAXIMIZING OBJLP;
33
34 SOLVE PORTQP USING NLP MINIMIZING OBJQP;
```

COMPILATION TIME = 1.210 SECONDS

VERID TP5-00-

038

GAMS 2.25 PC AT/XT

SIMPLE PORTFOLIO ONVESTMENT PROBLEM (MARKOWITZ)

Model Statistics SOLVE PORTLP USING LP FROM LINE 32

MODEL STATISTICS

BLOCKS OF EQUATIONS	2	SINGLE EQUATIONS	2
BLOCKS OF VARIABLES	4	SINGLE VARIABLES	4
NON ZERO ELEMENTS	7		

GENERATION TIME = 1.040 SECONDS

EXECUTION TIME = 1.970 SECONDS

VERID TP5-00-

038

GAMS 2.25 PC AT/XT

SIMPLE PORTFOLIO INVESTMENT PROBLEM (MARKOWITZ)

Solution Report SOLVE PORTLP USING LP FROM LINE 32

S O L V E S U M M A R Y

MODEL	PORTLP	OBJECTIVE	OBJLP
TYPE	LP	DIRECTION	MAXIMIZE
SOLVER	BDMLP	FROM LINE	32

\*\*\*\* SOLVER STATUS            1 NORMAL COMPLETION  
\*\*\*\* MODEL STATUS            1 OPTIMAL  
\*\*\*\* OBJECTIVE VALUE            1.2750

RESOURCE USAGE, LIMIT	1.270	1000.000
ITERATION COUNT, LIMIT	1	1000

BDM - LP      VERSION 1.01

A. Brooke, A. Drud, and A. Meeraus,  
Analytic Support Unit,  
Development Research Department,  
World Bank,  
Washington D.C. 20433, U.S.A.

D E M O N S T R A T I O N   M O D E

You do not have a full license for this program.  
The following size restrictions apply:  
Total nonzero elements: 1000

Estimate work space needed	--	33 Kb
Work space allocated	--	231 Kb

EXIT -- OPTIMAL SOLUTION FOUND.

	LOWER	LEVEL	UPPER	MARGINAL
---- EQU LP	.	.	.	1.000
---- EQU E2	1.000	1.000	1.000	1.200

	LOWER	LEVEL	UPPER	MARGINAL
---- VAR IBM	.	0.750	0.750	0.100
---- VAR GM	.	0.250	0.750	.
---- VAR GOLD	.	.	0.750	-0.120
---- VAR OBJLP	-INF	1.275	+INF	.

\*\*\*\* REPORT SUMMARY :    0    NONOPT  
                              0    INFEASIBLE  
                              0    UNBOUNDED

GAMS 2.25 PC AT/XT  
SIMPLE PORTFOLIO INVESTMENT PROBLEM (MARKOWITZ)  
Model Statistics    SOLVE PORTQP USING NLP FROM LINE 34

MODEL STATISTICS

BLOCKS OF EQUATIONS	3	SINGLE EQUATIONS	3
BLOCKS OF VARIABLES	4	SINGLE VARIABLES	4
NON ZERO ELEMENTS	10	NON LINEAR N-Z	3
DERIVATIVE POOL	8	CONSTANT POOL	3
CODE LENGTH	95		

GENERATION TIME = 2.360 SECONDS

EXECUTION TIME = 3.510 SECONDS  
038

VERID TP5-00-

GAMS 2.25 PC AT/XT

SIMPLE PORTFOLIO INVESTMENT PROBLEM (MARKOWITZ)

Solution Report SOLVE PORTLP USING LP FROM LINE 34

### S O L V E S U M M A R Y

MODEL	PORTLP	OBJECTIVE	OBJLP
TYPE	LP	DIRECTION	MAXIMIZE
SOLVER	MINOS5	FROM LINE	34

\*\*\*\* SOLVER STATUS 1 NORMAL COMPLETION  
\*\*\*\* MODEL STATUS 2 LOCALLY OPTIMAL  
\*\*\*\* OBJECTIVE VALUE 0.4210

RESOURCE USAGE, LIMIT	3.129	1000.000
ITERATION COUNT, LIMIT	3	1000
EVALUATION ERRORS	0	0

MINOS 5.3 (Nov. 1990) Ver: 225-DOS-02

B.A. Murtagh, University of New South Wales  
and

P.E. Gill, W. Murray, M.A. Saunders and M.H. Wright  
Systems Optimization Laboratory, Stanford University.

### D E M O N S T R A T I O N M O D E

You do not have a full license for this program.

The following size restrictions apply:

Total nonzero elements: 1000

Nonlinear nonzero elements: 300

Estimate work space needed	--	39 Kb
Work space allocated	--	40 Kb

EXIT -- OPTIMAL SOLUTION FOUND

MAJOR ITNS, LIMIT	1
FUNOBJ, FUNCON CALLS	8
SUPERBASICS	1
INTERPRETER USAGE	.21
NORM RG / NORM PI	3.732E-17

	LOWER	LEVEL	UPPER	MARGINAL
---- EQU QP	.	.	.	1.000

----	EQU E1	1.150	1.150	+INF	1.216
----	EQU E2	1.000	1.000	1.000	-0.556
		LOWER	LEVEL	UPPER	MARGINAL
----	VAR IBM	.	0.183	0.750	.
----	VAR GM	.	0.248	0.750	EPS
----	VAR GOLD	.	0.569	0.750	.
----	VAR OBJLP	-INF	1.421	+INF	.

\*\*\*\* REPORT SUMMARY :   0   NONOPT  
                           0   INFEASIBLE  
                           0   UNBOUNDED  
                           0   ERRORS

GAMS 2.25 PC AT/XT  
 SIMPLE PORTFOLIO INVESTMENT PROBLEM (MARKOWITZ)  
 Model Statistics   SOLVE PORTQP USING NLP FROM LINE 34

EXECUTION TIME       =       1.090 SECONDS                    VERID TP5-00-038

USER:  CACHE DESIGN CASE STUDIES                            G911007-1447AX-TP5  
           GAMS DEMONSTRATION VERSION

\*\*\*\* FILE SUMMARY

INPUT    C:\GAMS\PORT.GMS  
           OUTPUT   C:\GAMS\PORT.LST

# ALGORITHMS FOR CONSTRAINED PROBLEMS

Motivation: Build on unconstrained methods wherever possible.

## Classification of Methods:

1. Penalty Function - currently obsolete
2. Successive Linear Programming - only useful for "mostly linear" problems
3. Successive Quadratic Programming - library implementations - generic
4. Reduced Gradient Methods - (with Restoration) - GRG2
5. Reduced Gradient Methods - (without Restoration) - MINOS

We will concentrate on algorithms for classes 3 to 5.

Evaluation: Compare performance on "typical problem," cite experience on process problems.

# SUCCESSIVE QUADRATIC PROGRAMMING (SQP)

## Motivation:

Take Kuhn-Tucker conditions, expand in Taylor series about current point.

Take Newton step (QP) to determine next point.

## Derivation

### Kuhn - Tucker Conditions

$$\begin{aligned}\nabla_{\mathbf{x}}L(\mathbf{x}^*, \mathbf{u}^*, \mathbf{v}^*) &= \\ \nabla f(\mathbf{x}^*) + \nabla \mathbf{g}(\mathbf{x}^*) \mathbf{u}^* + \nabla \mathbf{h}(\mathbf{x}^*) \mathbf{v}^* &= \mathbf{0} \\ \mathbf{g}_A(\mathbf{x}^*) &= \mathbf{0} \\ \mathbf{h}(\mathbf{x}^*) &= \mathbf{0}\end{aligned}$$

where  $\mathbf{g}_A$  are the active constraints.

### Newton - Step

$$\begin{bmatrix} \nabla_{\mathbf{xx}} L & \nabla \mathbf{g}_A & \nabla h \\ \nabla \mathbf{g}_A^T & \mathbf{0} & \mathbf{0} \\ \nabla h^T & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{x} \\ \Delta \mathbf{u} \\ \Delta \mathbf{v} \end{bmatrix} = - \begin{bmatrix} \nabla_{\mathbf{x}} L(\mathbf{x}^i, \mathbf{u}^i, \mathbf{v}^i) \\ \mathbf{g}_A(\mathbf{x}^i) \\ h(\mathbf{x}^i) \end{bmatrix}$$

Need:  $\nabla_{\mathbf{xx}}L$  calculated  
correct active set  $\mathbf{g}_A$   
good estimates of  $\mathbf{u}^i, \mathbf{v}^i$

# SQP Chronology

## 1. Wilson (1963)

- active set can be determined by solving QP:

$$\begin{aligned} \text{Min } & \nabla f(\mathbf{x}^i)^T \mathbf{d} + 1/2 \mathbf{d}^T \nabla_{\mathbf{xx}} L(\mathbf{x}^i, \mathbf{u}^i, \mathbf{v}^i) \mathbf{d} \\ & \mathbf{d} \\ \text{s.t. } & \mathbf{g}(\mathbf{x}^i) + \nabla \mathbf{g}(\mathbf{x}^i)^T \mathbf{d} \leq 0 \\ & \mathbf{h}(\mathbf{x}^i) + \nabla \mathbf{h}(\mathbf{x}^i)^T \mathbf{d} = 0 \end{aligned}$$

## 2. Han (1976), (1977), Powell (1977), (1978)

- estimate  $\nabla_{\mathbf{xx}} L$  using a quasi-Newton update (BFGS) to form  $\mathbf{B}^i$  (positive definite)
- use a line search to converge from poor starting points.

### Notes:

- Similar methods were derived using penalty (not Lagrange) functions.
- Method converges quickly; very few function evaluations.
- Not well suited to large problems (full space update used). For  $n > 100$ , say, use reduced space methods (e.g. MINOS).

# Improvements to SQP

1. What about  $\nabla_{xx}L$ ?

- need to get second derivatives for  $f(x)$ ,  $g(x)$ ,  $h(x)$ .
- need to estimate multipliers,  $u^i$ ,  $v^i$ ;  $\nabla_{xx}L$  may not be positive semidefinite

⇒

Approximate  $\nabla_{xx}L(x^i, u^i, v^i)$  by  $B^i$ , a symmetric positive definite matrix.

$$B^{i+1} = B^i - \frac{B^i s s^T B^i}{s^T B^i s} + \frac{y y^T}{s^T y}$$

BFGS Formula  $s = x^{i+1} - x^i$   
 $y = \nabla L(x^{i+1}, u^{i+1}, v^{i+1}) - \nabla L(x^i, u^i, v^i)$

- second derivatives approximated by differences in first derivatives.
- $B^i$  ensures *unique* QP solution

2. How do we know  $g_A$ ?

- Put all  $g(x)$ 's into QP and let QP determine constraint activity
- At each iteration,  $k$ , solve:

$$\begin{aligned} & \text{Min } \nabla f(x^i)^T d + 1/2 d^T B^i d \\ & \text{s.t. } g(x^i) + \nabla g(x^i)^T d \leq 0 \\ & \quad h(x^i) + \nabla h(x^i)^T d = 0 \end{aligned}$$

### 3. Convergence from poor starting points

- Like Newton's method, choose  $\alpha$  (stepsize) to ensure progress toward optimum:  $x^{i+1} = x^i + \alpha d$ .
- $\alpha$  is chosen by making sure a *merit function* is decreased at each iteration.

#### *Exact Penalty Function*

$$\psi(x) = f(x) + \mu [\sum \max(0, g_j(x)) + \sum |h_j(x)|]$$
$$\mu > \max_j \{ |u_j|, |v_j| \}$$

#### *Augmented Lagrange Function*

$$\psi(x) = f(x) + u^T g(x) + v^T h(x)$$
$$+ \eta/2 \{ \sum (h_j(x))^2 + \sum \max(0, g_j(x))^2 \}$$

## NEWTON-LIKE PROPERTIES

Fast (Local) Convergence

$$B = \nabla_{xx}L$$

Quadratic

B and  $\nabla_{xx}L$  both p.d

Superlinear

B is p.d + Q-N update

2 step Superlinear

Can enforce Global Convergence

Ensure decrease of merit function by taking  $\alpha \leq 1$

Trust regions provide a stronger guarantee of global convergence - but harder to implement.

## BASIC SQP ALGORITHM

0. Guess  $x^0$ , Set  $B^0 = I$  (Identity)  
Evaluate  $f(x^0)$ ,  $g(x^0)$  and  $h(x^0)$ .
1. At  $x^i$ , evaluate  $\nabla f(x^i)$ ,  $\nabla g(x^i)$ ,  $\nabla h(x^i)$ .
2. If  $i > 0$ , update  $B^i$  using the BFGS Formula.
3. Solve:  
$$\text{Min}_d \nabla f(x^i)^T d + 1/2 d^T B^i d$$
$$\text{s.t. } g(x^i) + \nabla g(x^i)^T d \leq 0$$
$$h(x^i) + \nabla h(x^i)^T d = 0$$

If Kuhn-Tucker error is less than tolerance:

e.g.,  $|\nabla L^T d| \leq |\nabla f^T d| + \sum |u_j g_j| + \sum |v_j h_j| \leq \varepsilon$   
STOP, else go to 4.

4. Find  $\alpha$  so that  $0 < \alpha \leq 1$  and  $\Psi(x^i + \alpha d) < \Psi(x^i)$   
sufficiently (Each trial requires evaluation of  $f(x)$ ,  $g(x)$   
and  $h(x)$ ).
5.  $x^{i+1} = x^i + \alpha d$ . Set  $i = i + 1$  Go to 1.

## PROBLEMS WITH SQP

Nonsmooth Functions

- Reformulate

Ill-conditioning

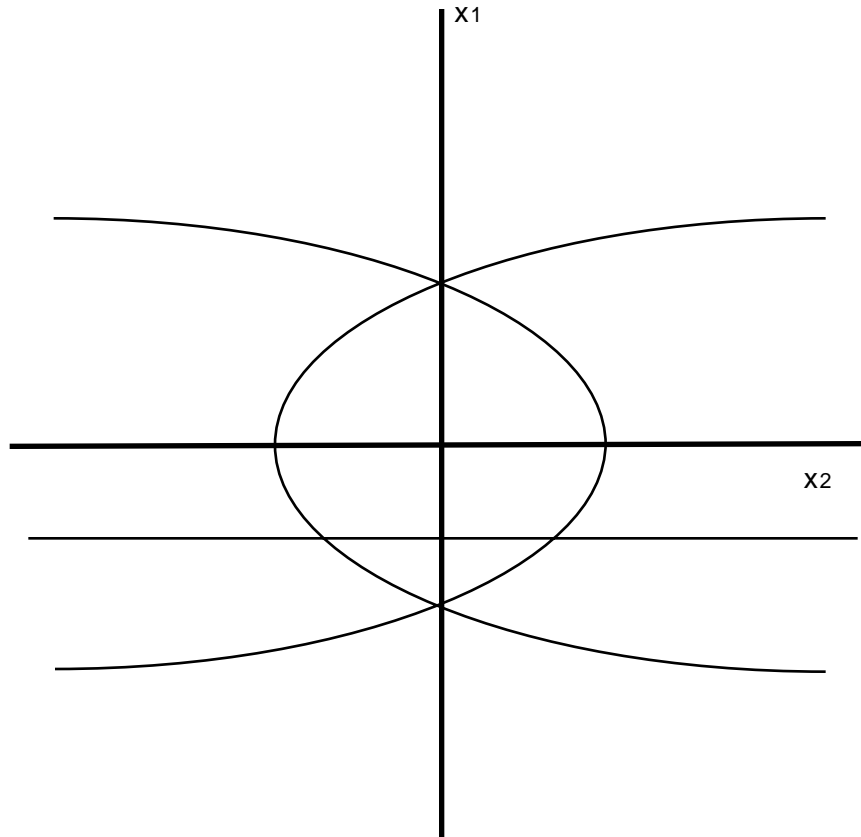
- Proper scaling

Poor Starting Points

- Global convergence can help (Trust regions)

Inconsistent Constraint Linearizations

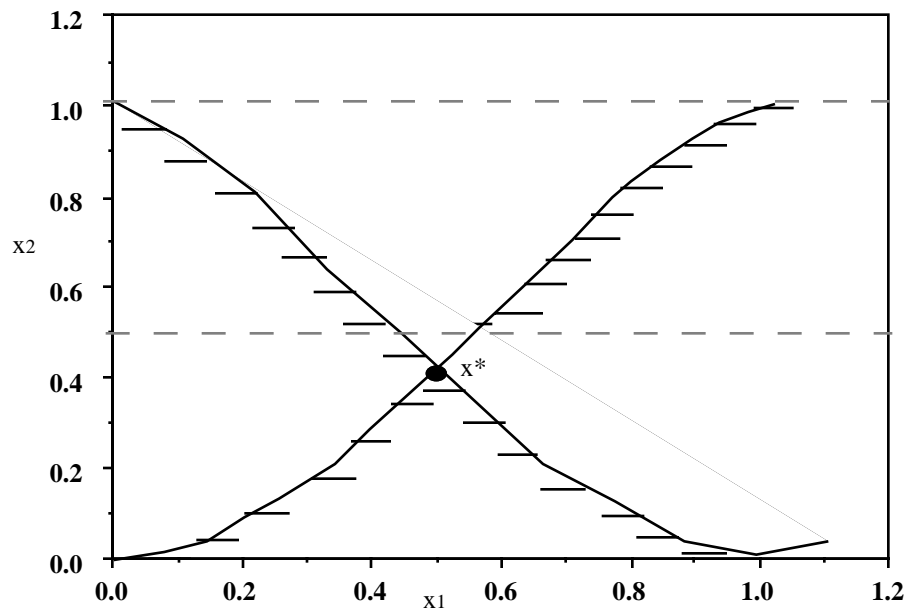
- Can lead to infeasible QP's



$$\begin{aligned} & \text{Min } x_2 \\ \text{s.t. } & 1 + x_1 - (x_2)^2 \leq 0 \\ & 1 - x_1 - (x_2)^2 \leq 0 \\ & x_2 \geq -1/2 \end{aligned}$$

# SUCCESSIVE QUADRATIC PROGRAMMING (SQP)

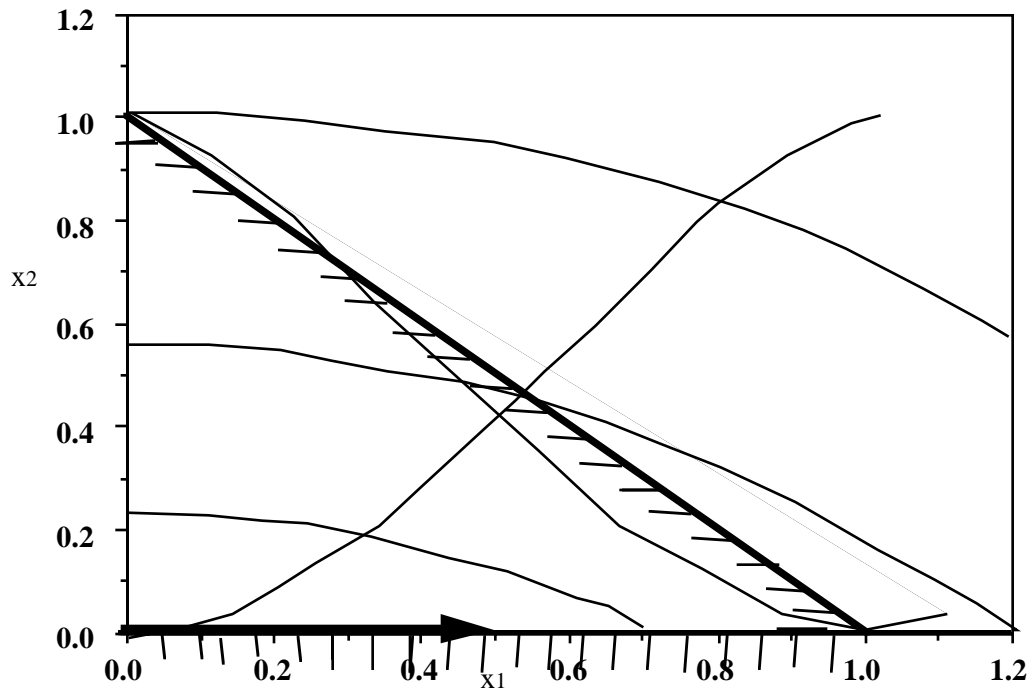
## Test Problem



$$\begin{aligned} & \text{Min } x_2 \\ \text{s.t. } & -x_2 + 2x_1^2 - x_1^3 \leq 0 \\ & -x_2 + 2(1-x_1)^2 - (1-x_1)^3 \leq 0 \\ & x^* = [0.5, 0.375]. \end{aligned}$$

# SUCCESSIVE QUADRATIC PROGRAMMING

## 1st ITERATION

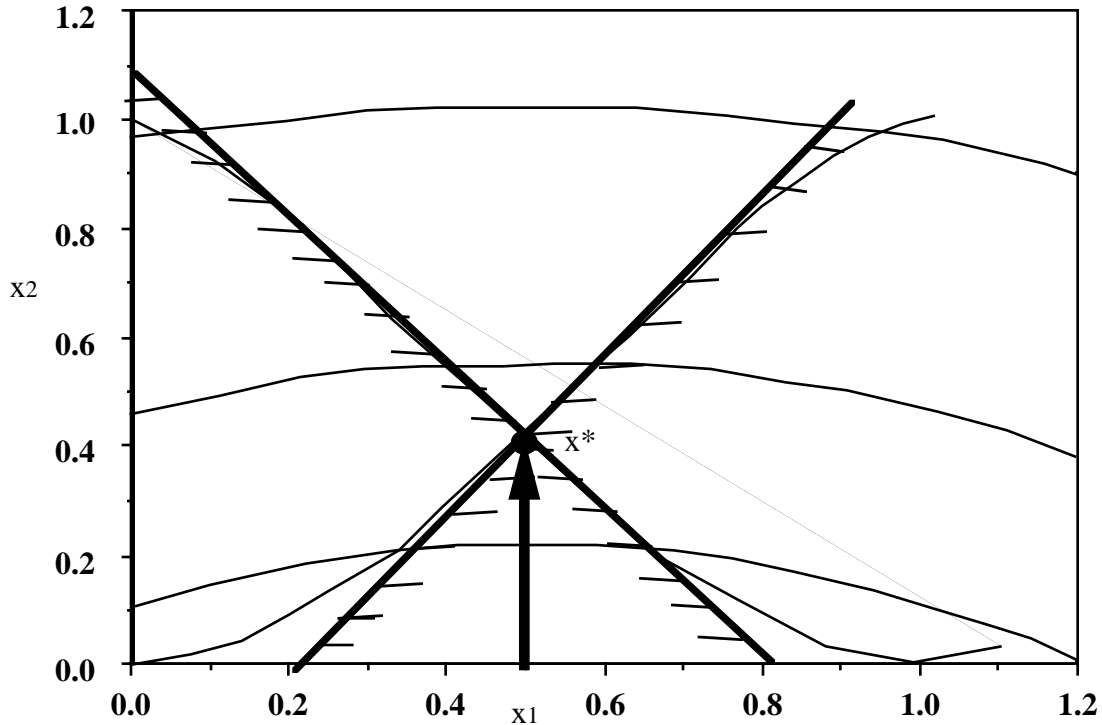


Start from the origin ( $x^0 = [0, 0]^T$ ) with  $B^0 = I$ , form:

$$\begin{aligned} \text{Min} \quad & d_2 + 1/2 (d_1^2 + d_2^2) \\ \text{s.t.} \quad & d_2 \geq 0 \\ & d_1 + d_2 \geq 1 \\ & d = [1, 0]^T \text{ with } \mu_1 = 0 \text{ and } \mu_2 = 1. \end{aligned}$$

# SUCCESSIVE QUADRATIC PROGRAMMING

## 2nd ITERATION



From  $x^1 = [0.5, 0]^T$  with  $B^1 = I$  (no update from BFGS possible), form:

$$\begin{aligned} \text{Min} \quad & d_2 + \frac{1}{2} (d_1^2 + d_2^2) \\ \text{s.t.} \quad & -1.25 d_1 - d_2 + 0.375 \leq 0 \\ & 1.25 d_1 - d_2 + 0.375 \leq 0 \\ & d = [0, 0.375]^T \text{ with } \mu_1 = 0.5 \text{ and } \mu_2 = 0.5 \end{aligned}$$

$x^2 = [0.5, 0.375]^T$  is optimal

## REDUCED GRADIENT METHOD WITH RESTORATION (GRG)

### Motivation:

- Decide which constraints are active or inactive, partition variables as dependent and independent.
- Solve unconstrained problem in space of independent variables.

### Derivation:

$$\begin{aligned} & \text{Min } f(x) \\ & \text{s.t. } g(x) + s = 0 \quad (\text{add slack variable}) \\ & \quad h(x) = 0 \\ & \quad a' \leq x \leq b', s \geq 0 \end{aligned}$$

$$\begin{aligned} \Rightarrow & \text{Min } f(z) \\ & \quad h(z) = 0 \\ & \quad a \leq z \leq b \end{aligned}$$

Let  $z^T = [z_I^T \ z_D^T]$  to optimize wrt  $z_I$  with  $h(z_I, z_D) = 0$  we need a constrained derivative or reduced gradient wrt  $z_I$ . Dependent variables are  $z_D \in \mathbb{R}^m$

Note:  $h(z_I, z_D) = 0 \Rightarrow z_D = \psi(z_I)$

Define reduced gradient:

$$\frac{df}{dz_I} = \frac{\partial f}{\partial z_I} + \frac{dz_D}{dz_I} \frac{\partial f}{\partial z_D}$$

Now  $h(z_I, z_D) = 0$ , and

$$dh = \left( \frac{\partial h}{\partial z_I} \right)^T dz_I + \left( \frac{\partial h}{\partial z_D} \right)^T dz_D = 0$$

and 
$$\frac{dz_D}{dz_I} = - \left( \frac{\partial h}{\partial z_I} \right) \left( \frac{\partial h}{\partial z_D} \right)^{-1} = \nabla_{z_I} h \left( \nabla_{z_D} h \right)^{-1}$$

$$\frac{df}{dz_I} = - \frac{\partial f}{\partial z_I} - \nabla_{z_I} h \left( \nabla_{z_D} h \right)^{-1} \frac{\partial f}{\partial z_D}$$

Note: By remaining feasible always,  $h(z) = 0$ ,  $a \leq z \leq b$ , one can apply an unconstrained algorithm (quasi-Newton)

using  $\frac{df}{dz_I}$ .

Solve problem in reduced space (n-m) of variables.

## EXAMPLE OF REDUCED GRADIENT

$$\begin{array}{ll} \text{Min} & x_1^2 - 2x_2 \\ \text{s.t.} & 3x_1 + 4x_2 = 24 \end{array}$$

$$\nabla h^T = [3 \quad 4] \quad \nabla f = \begin{bmatrix} 2x_1 \\ -2 \end{bmatrix}$$

Let

$$\left. \begin{array}{l} z = x \\ z_I = x_1 \\ z_D = x_2 \end{array} \right\} \frac{df}{dz_I} = \frac{\partial f}{\partial z_I} - \nabla_{z_I} h (\nabla_{z_I} h)^{-1} \frac{\partial f}{\partial z_D}$$

$$\frac{df}{dx_I} = \frac{\partial f}{\partial x_1} - \left( \frac{\partial h}{\partial x_1} \right) \left( \frac{\partial h}{\partial x_2} \right)^{-1} \frac{\partial f}{\partial x_2}$$

$$\frac{df}{dx_1} = 2x_1 - \left( \frac{3}{4} \right) (-2) = 2x_1 + \frac{3}{2}$$

Note:  $\nabla h^T$  is  $m \times n$ ;  $\nabla_{z_I} h^T$  is  $m \times (n-m)$ ;  $\nabla_{z_I} h$  is  $m \times m$

$\frac{df}{dz_I}$  is change in  $f$  along constraint direction per unit change in  $z_I$

## SKETCH OF GRG ALGORITHM (Lasdon)

1. Starting with  $\text{Min } f(z)$ , partition  
s.t.  $h(z) = 0$   
 $a \leq z \leq b$

$$z^T = [z_I^T, z_D^T] \text{ with } z_I \in \mathbb{R}^{n-\text{meq}}, z_D \in \mathbb{R}^{n-\text{meq}}$$

{Choose  $z_D$  so that they are not close to their bounds.}

2. Choose  $z^0$ , so that  $h(z^0) = 0$   $z_D$  so that they are not close to their bounds.
3. Calculate  $df/dz_I$  at  $z^i$  and  $d_I = - (B^i)^{-1} \frac{df}{dz_I}$ . ( $B^i$  can be from a Q-N update formula.) If  $(z_I + d_I)_j$  violates bound, set  $d_{Ij} = 0$ .

If  $\left\| \frac{df}{dz_I} \right\| \leq \epsilon$ , and  $\|d_I\| \leq \epsilon$ , stop. Else, continue with 4.

4. Perform a line search to get  $\alpha$ .
- For each  $\bar{z}_I = z_I^i + \alpha d_I$ , solve for  $z_D$ , i.e.
$$z_D^{k+1} = z_D^k - (\nabla_D h(\bar{z}_I, z_D^k))^T)^{-1} h(\bar{z}_I, z_D^k)$$
  - Reduce  $\alpha$  if  $z_D$  violates bound or  $f(\bar{z}) \geq f(z^i)$ .

Notes:

1. GRG (or GRG2) has been implemented on PC's as GINO and is very reliable and robust. It is also the optimization solver in MS EXCEL.
2. It uses Q-N (DFP) for small problems but can switch to conjugate gradients if problem gets large.
3. Convergence of  $h(z_D, z_I) = 0$  can get very expensive for flowsheeting problems because  $\nabla h$  is required.
4. Safeguards can be added so that restoration (step 4.) can be dropped and efficiency increases.

## REDUCED GRADIENT METHOD WITHOUT RESTORATION (MINOS/Augmented)

Motivation: Efficient algorithms are available that solve linearly constrained optimization problems (MINOS):

$$\begin{aligned} \text{Min } & f(x) \\ \text{s.t. } & Ax \leq b \\ & Cx = d \end{aligned}$$

Extend to nonlinear problems, through successive linearization.

Strategy: (Robinson, Murtagh & Saunders)

1. Partition variables into independent,  $z_I$  and dependent variables,  $z_D$ .

2. Linearize active constraints about starting point,  
 $z \Rightarrow Dz = c$ .

3. Let  $\Psi = f(z) + v^T h(z) + \beta (h^T h)$  (Augmented Lagrange) and solve linearly constrained problem:

$$\begin{aligned} \text{Min } & \Psi(z) \\ \text{s.t. } & Dz = c \\ & a \leq z \leq b \end{aligned}$$

using reduced gradients (as in GRG) to get  $z'$ .

4. Linearize constraints about  $z'$  and go to 3.

5. Algorithm terminates when no movement occurs between steps 3) and 4).

Notes:

1. MINOS has been implemented very efficiently to take care of linearity. It becomes LP Simplex method if problem is totally linear. Also, very efficient matrix routine.
2. Although no restoration takes place, constraint nonlinearities are reflected in  $\Psi(z)$  during step 3). Hence MINOS is more efficient than GRG.
3. Major iterations (steps 3) - 4)) converge at a quadratic rate.

## RECOMMENDATIONS FOR CONSTRAINED OPTIMIZATION

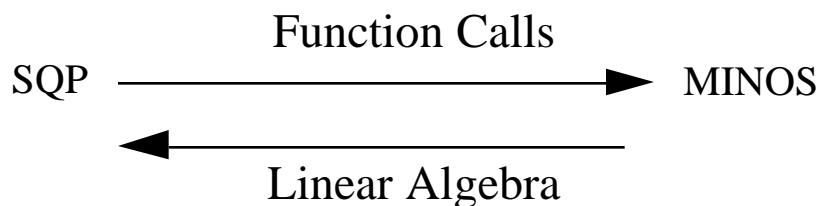
1. Best current algorithms

- GRG 2/CONOPT
- MINOS
- SQP

2. GRG 2 (or CONOPT) is generally slower, but is robust. Use with highly nonlinear functions. Solver in Excel!

3. For small problems ( $n \leq 100$ ) with nonlinear constraints, use SQP.

4. For large problems ( $n \geq 100$ ) with mostly linear constraints, use MINOS. ==> Difficulty with many nonlinearities



Small, Nonlinear Problems - SQP (generic) solves QP's, not LCNLP's, fewer function calls.

Large, Mostly Linear Problems - MINOS performs sparse constraint decomposition. Works efficiently in reduced space if function calls are cheap!

# AVAILABLE SOFTWARE FOR CONSTRAINED OPTIMIZATION

## NaG Routines

### *Unconstrained Optimization*

E04CCF - Unconstrained minimum, simplex algorithm, function of several variables using function values only (comprehensive)

E04DGF - Unconstrained minimum, preconditioned conjugate gradient algorithm, function of several variables using first derivatives (comprehensive)

E04FCF - Unconstrained minimum of a sum of squares, combined Gauss--Newton and modified Newton algorithm using function values only (comprehensive)

E04FYF - Unconstrained minimum of a sum of squares, combined Gauss--Newton and modified Newton algorithm using function values only (easy-to-use)

E04GBF - Unconstrained minimum of a sum of squares, combined Gauss--Newton and quasi-Newton algorithm using first derivatives (comprehensive)

E04GDF - Unconstrained minimum of a sum of squares, combined Gauss--Newton and modified Newton algorithm using first derivatives (comprehensive)

E04GYF - Unconstrained minimum of a sum of squares, combined Gauss--Newton and quasi-Newton algorithm, using first derivatives (easy-to-use)

E04GZF - Unconstrained minimum of a sum of squares, combined Gauss--Newton and modified Newton algorithm using first derivatives (easy-to-use)

E04HEF - Unconstrained minimum of a sum of squares, combined Gauss--Newton and modified Newton algorithm, using second derivatives (comprehensive)

E04HYF - Unconstrained minimum of a sum of squares, combined Gauss--Newton and modified Newton algorithm, using second derivatives (easy-to-use)

E04JYF - Minimum, function of several variables, quasi-Newton algorithm, simple bounds, using function values only (easy-to-use)

E04KDF - Minimum, function of several variables, modified Newton algorithm, simple bounds, using first derivatives (comprehensive)

E04KYF - Minimum, function of several variables, quasi-Newton algorithm, simple bounds, using first derivatives (easy-to-use)

E04KZF - Minimum, function of several variables, modified Newton algorithm, simple bounds, using first derivatives (easy-to-use)

E04LBF - Minimum, function of several variables, modified Newton algorithm, simple bounds, using first and second derivatives (comprehensive)

E04LYF - Minimum, function of several variables, modified Newton algorithm, simple bounds, using first and second derivatives (easy-to-use)

### ***Specialized Constrained Algorithms***

E04MFF - LP problem (dense)

E04NCF - Convex QP problem or linearly-constrained linear least-squares problem (dense)

### ***SQP Routines***

E04UCF - Minimum, function of several variables, sequential QP method, nonlinear constraints, using function values and optionally first derivatives (forward communication, comprehensive)

E04UFF - Minimum, function of several variables, sequential QP method, nonlinear constraints, using function values and optionally first derivatives (reverse communication, comprehensive)

E04UNF - Minimum of a sum of squares, nonlinear constraints, sequential QP method, using function values and optionally first derivatives (comprehensive)

## **GAMS Programs**

CONOPT - Generalized Reduced Gradient method with restoration

MINOS - Generalized Reduced Gradient method with restoration

## **MS Excel**

Solver uses Generalized Reduced Gradient method with restoration

## RULES FOR FORMULATING NONLINEAR PROGRAMS

1) Avoid overflows and undefined terms, (do not divide, take logs, etc.)

$$\text{e.g. } x + y - \ln x = 0$$

↓

$$x + y - u = 0$$

$$\exp u - z = 0$$

2) If constraints must always be enforced, make sure they are linear or bounds.

$$\text{e.g. } \sqrt{xy - z^2} = 3$$

↓

$$vu = 3$$

$$u^2 - (xy - z^2) = 0$$

$$u \geq 0$$

3) Exploit linear constraints as much as possible.

e.g. mass balance

$$x_i L + y_i V = F$$

↓

$$l_i + v_i = f_i$$

$$L - \sum l_i = 0, \text{ etc.}$$

4) Use bounds and constraints to enforce characteristic solutions.

$$\text{e.g. } a \leq x \leq b$$

$$g(x) \leq 0$$

to isolate correct root of  $h(x) = 0$ .

5) Exploit global properties when possibility exists.

Is problem convex (no nonlinear equations)?

- Linear Program
- Quadratic Program

Is the problem a Geometric Program (logarithmic transformation converts to a convex problem)?

6) Exploit problem structure when possible.

e.g. Min  $[Tx - 3Ty]$

$$\text{s.t. } xT + y - T^2y = 5$$

$$4x - 5Ty + Tx = 7$$

$$0 \leq T \leq 1$$

(If T is fixed  $\Rightarrow$  solve LP)

$\Rightarrow$  put T in outer optimization loop.