Nonlinear Programming: Concepts, Algorithms and Applications

L. T. Biegler
Chemical Engineering Department
Carnegie Mellon University
Pittsburgh, PA

Nonlinear Programming and Process Optimization

Introduction

Unconstrained Optimization
- Algorithms
- Newton Methods
- Quasi-Newton Methods

Constrained Optimization
- Karush Kuhn-Tucker Conditions
- Special Classes of Optimization Problems
- Reduced Gradient Methods (GRG2, CONOPT, MINOS)
- Successive Quadratic Programming (SQP)
- Interior Point Methods (IPOPT)

Process Optimization
- Black Box Optimization
- Modular Flowsheet Optimization – Infeasible Path
- The Role of Exact Derivatives

Large-Scale Nonlinear Programming
- rSQP: Real-time Process Optimization
- IPOPT: Blending and Data Reconciliation

Further Applications
- Sensitivity Analysis for NLP Solutions
- Multi-Scenario Optimization Problems

Summary and Conclusions

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!

Optimization 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., D.M. Himmelblau, and L. S. Lasdon, Optimization of Chemical Processes, McGraw-Hill, 2001.

2. Papalambros, P. and D. Wilde, Principles of Optimal Design. Cambridge Press, 1988.

3. Reklaitis, G., 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), SIAM (1995)

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, 2007

Motivation

Scope of optimization

Provide *systematic framework* for searching among a specified space of alternatives to identify an “optimal” design, i.e., as a *decision-making tool*

Premise

Conceptual formulation of optimal product and process design corresponds to a mathematical programming problem

MINLP → NLP

$\leq$

$\in \quad \in$

Optimization in Design, Operations and Control

	MILP	MINLP	Global	LP,QP	NLP	SA/GA
HENS	x	x	x	x	x	x
MENS	x	x	x	x	x	x
Separations	x	x				
Reactors		x	x	x	x	
Equipment Design		x			x	x
Flowsheeting		x			x	
Scheduling	x	x		x		x
Supply Chain	x	x		x		
Real-time optimization				x	x	
Linear MPC				x		
Nonlinear MPC			x		x	
Hybrid	x				x	

Unconstrained Multivariable Optimization

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

Equivalent to: Max $-f(x)$, $x \in R^n$

Nonsmooth Functions
- Direct Search Methods
- Statistical/Random Methods

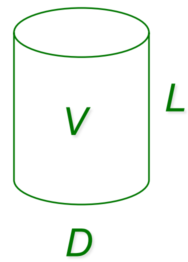
Smooth Functions
- 1st Order Methods
- *Newton Type Methods*
- Conjugate Gradients

Example: Optimal Vessel Dimensions

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

Constrained Problem

(1)



Convert to Unconstrained (Eliminate L)

$$\text{Min}\left\{ C_T \frac{\pi D^2}{2} + C_S \frac{4V}{D} = \text{cost} \right\}$$

$$\frac{d(\text{cost})}{dD} = C_T \pi D - \frac{4VC_s}{D^2} = 0 \qquad (2)$$

$$D = \left(\frac{4V}{\pi} \frac{C_S}{C_T} \right)^{1/3} \qquad L = \left(\frac{4V}{\pi} \right)^{1/3} \left(\frac{C_T}{C_S} \right)^{2/3}$$

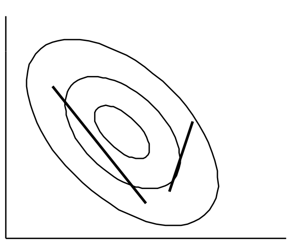
==> $L/D = C_T/C_S$

Note:

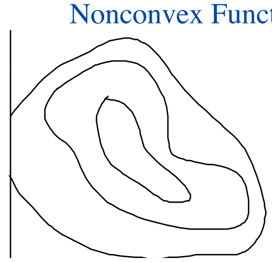
- What if L cannot be eliminated in (1) explicitly? (strange shape)
- What if D cannot be extracted from (2)? (cost correlation implicit)

Two Dimensional Contours of F(x)

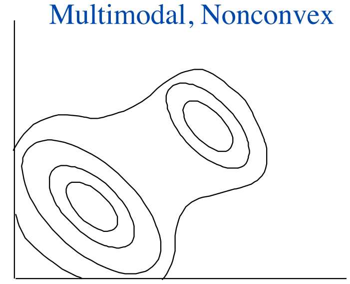
Convex Function

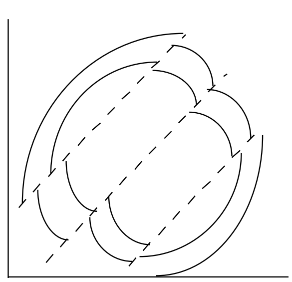


Nonconvex Function

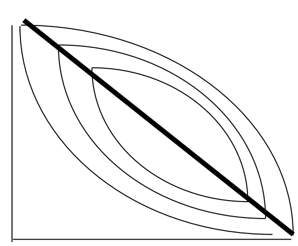


Multimodal, Nonconvex





Discontinuous



Nondifferentiable (convex)

Local vs. Global Solutions

Convexity Definitions

•a set (region) **X** is convex, if and only if it satisfies:

$$\alpha y + (1-\alpha)z \in X$$

for all α, $0 \leq \alpha \leq 1$, for all points y and z in **X**.

• *f(x)* is convex in domain **X**, if and only if it satisfies:

$$f(\alpha y + (1-\alpha) z) \leq \alpha f(y) + (1-\alpha)f(z)$$

for any α, $0 \leq \alpha \leq 1$, at all points y and z in **X**.

•Find a *local minimum* point x* for f(x) for feasible region defined by constraint functions: *$f(x^*) \leq f(x)$ for all x satisfying the constraints in some neighborhood around x^** *(not for all $x \in \mathbf{X}$)*

•Sufficient condition for a local solution to the NLP to be a global is that $f(x)$ is convex for $x \in \mathbf{X}$.

•Finding and verifying *global solutions* will not be considered here.

•Requires a more expensive search (e.g. spatial branch and bound).

Linear Algebra - Background

Some Definitions

- Scalars - Greek letters, α, β, γ
- Vectors - Roman Letters, lower case
- Matrices - Roman Letters, upper case
- Matrix Multiplication:
  $C = A B$ if $A \in \Re^{n \times m}$, $B \in \Re^{m \times p}$ and $C \in \Re^{n \times p}$, $C_{ij} = \Sigma_k A_{ik} B_{kj}$
- Transpose - if $A \in \Re^{n \times m}$,
  interchange rows and columns --> $A^T \in \Re^{m \times n}$
- Symmetric Matrix - $A \in \Re^{n \times n}$ (square matrix) and $A = A^T$
- Identity Matrix - I, square matrix with ones on diagonal
  and zeroes elsewhere.
- Determinant: "Inverse Volume" measure of a square matrix
  $\det(A) = \Sigma_i (-1)^{i+j} A_{ij} \underline{A}_{ij}$ for any j, or
  $\det(A) = \Sigma_j (-1)^{i+j} A_{ij} \underline{A}_{ij}$ for any i, where $\underline{A}_{ij}$ is the determinant of an order *n-1* matrix with row i and column j removed.
  $\det(I) = 1$
- Singular Matrix: $\det(A) = 0$

Linear Algebra - Background

Gradient Vector - ($\nabla f(x)$)

$$\nabla f = \begin{bmatrix} \partial f / \partial x_1 \\ \partial f / \partial x_2 \\ \ldots\ldots \\ \partial f / \partial x_n \end{bmatrix}$$

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

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

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

Linear Algebra - Background

- Some Identities for Determinant
  $\det(A\,B) = \det(A)\det(B)$; $\det(A) = \det(A^T)$
  $\det(\alpha A) = \alpha^n \det(A)$; $\det(A) = \prod_i \lambda_i(A)$

- Eigenvalues: $\det(A - \lambda I) = 0$, Eigenvector: $Av = \lambda v$
  Characteristic values and directions of a matrix.
  For nonsymmetric matrices eigenvalues can be complex,
  so we often use singular values, $\sigma = \lambda(A^T A)^{1/2} \geq 0$

- Vector Norms
  $\| x \|_p = \{\Sigma_i |x_i|^p\}^{1/p}$
  (most common are $p = 1$, $p = 2$ (Euclidean) and $p = \infty$ (max norm = $\max_i |x_i|$))
- Matrix Norms
  $\|A\| = \max \|A x\|/\|x\|$ over x (for p-norms)
  $\|A\|_1$ - max column sum of A, $\max_j (\Sigma_i |A_{ij}|)$
  $\|A\|_\infty$ - maximum row sum of A, $\max_i (\Sigma_j |A_{ij}|)$
  $\|A\|_2 = [\sigma_{max}(A)]$ (spectral radius)
  $\|A\|_F = [\Sigma_i \Sigma_j (A_{ij})^2]^{1/2}$ (Frobenius norm)
  $\kappa(A) = \|A\| \|A^{-1}\|$ (condition number) $= \sigma_{max}/\sigma_{min}$ (using 2-norm)

Linear Algebra - Eigenvalues

Find v and λ where $Av_i = \lambda_i v_i$, i = i,n

Note: $Av - \lambda v = (A - \lambda I) v = 0$ or $\det (A - \lambda I) = 0$

For this relation λ is an eigenvalue and v is an eigenvector of A.

If A is symmetric, all λ_i are real

- $\lambda_i > 0$, i = 1, n; A is positive definite
- $\lambda_i < 0$, i = 1, n; A is negative definite
- $\lambda_i = 0$, some i: A is singular

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

$x^T Ax \quad \Rightarrow \quad A V = V \Lambda$

V - eigenvector matrix (n x n)

Λ - eigenvalue (diagonal) matrix = $\mathrm{diag}(\lambda_i)$

If A is symmetric, all λ_i are real and V can be chosen orthonormal ($V^{-1} = V^T$).

Thus, $A = V \Lambda V^{-1} = V \Lambda V^T$

For Quadratic Function: $Q(x) = a^T x + \frac{1}{2} x^T Ax$

Define: $z = V^T x$ and $Q(Vz) = (a^T V) z + \frac{1}{2} z^T (V^T AV) z$

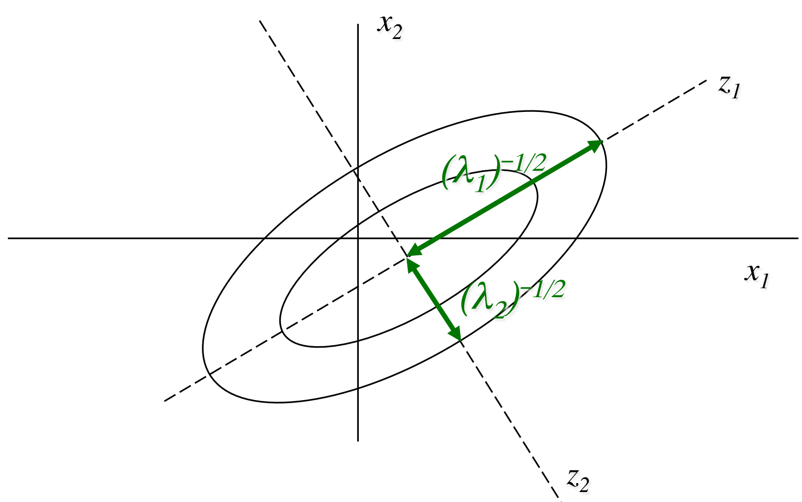
$= (a^T V) z + \frac{1}{2} z^T \Lambda z$

Minimum occurs at (if $\lambda_i > 0$) $x = -A^{-1} a$ or $x = Vz = -V(\Lambda^{-1} V^T a)$

Positive (Negative) Curvature Positive (Negative) Definite Hessian

Both eigenvalues are strictly positive (negative)

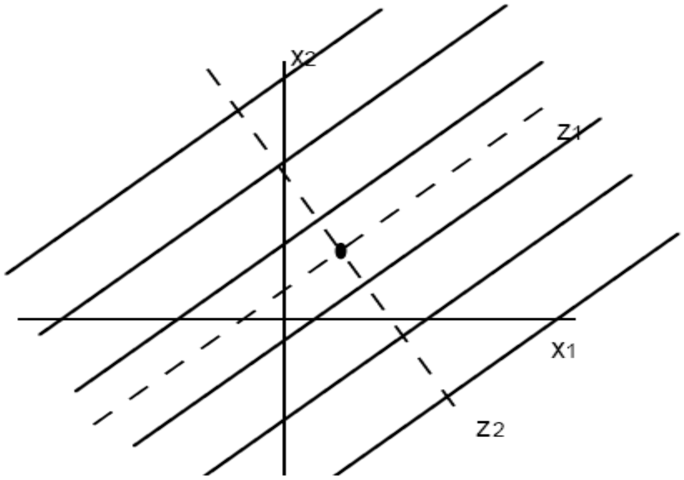
- A is positive (negative) definite
- Stationary points are minima (maxima)



Zero Curvature
Singular Hessian

One eigenvalue is zero, the other is strictly positive or negative

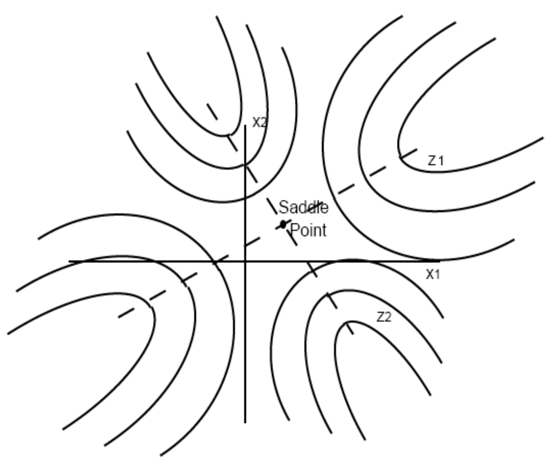
- A is positive semidefinite or negative semidefinite
- There is a ridge of stationary points (minima or maxima)



Indefinite Curvature
Indefinite Hessian

One eigenvalue is positive, the other is negative

- Stationary point is a saddle point
- A is indefinite



Note: these can also be viewed as two dimensional projections for higher dimensional problems

Eigenvalue Example

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

$$AV = V\Lambda \quad \text{with } A = \begin{bmatrix}2 & 1\\1 & 2\end{bmatrix}$$

$$V^T AV = \Lambda = \begin{bmatrix}1 & 0\\0 & 3\end{bmatrix} \text{ with V } = \begin{bmatrix}1/\sqrt{2} & 1/\sqrt{2}\\-1/\sqrt{2} & 1/\sqrt{2}\end{bmatrix}$$

- **All eigenvalues are positive**
- **Minimum occurs at $z^* = -\Lambda^{-1} V^T a$**

$$z = V^T x = \begin{bmatrix}(x_1 - x_2)/\sqrt{2}\\(x_1 + x_2)/\sqrt{2}\end{bmatrix} \qquad x = Vz = \begin{bmatrix}(x_1 + x_2)/\sqrt{2}\\(-x_1 + x_2)/\sqrt{2}\end{bmatrix}$$

$$z^* = \begin{bmatrix}0\\-2/(3\sqrt{2})\end{bmatrix} \qquad x^* = \begin{bmatrix}-1/3\\-1/3\end{bmatrix}$$

Comparison of Optimization Methods

1. Convergence Theory

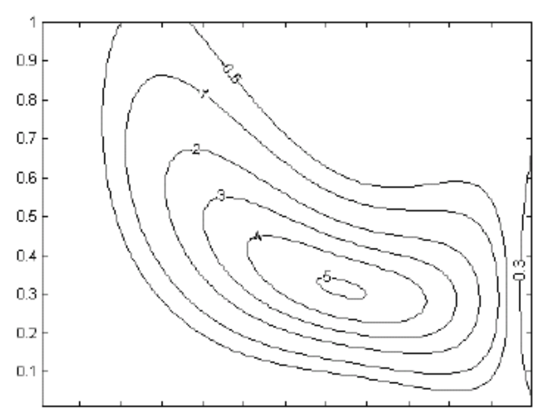
- Global Convergence - will it converge to a local optimum (or stationary point) from a poor starting point?
- Local Convergence Rate - how fast will it converge close to this point?

2. Benchmarks on Large Class of Test Problems

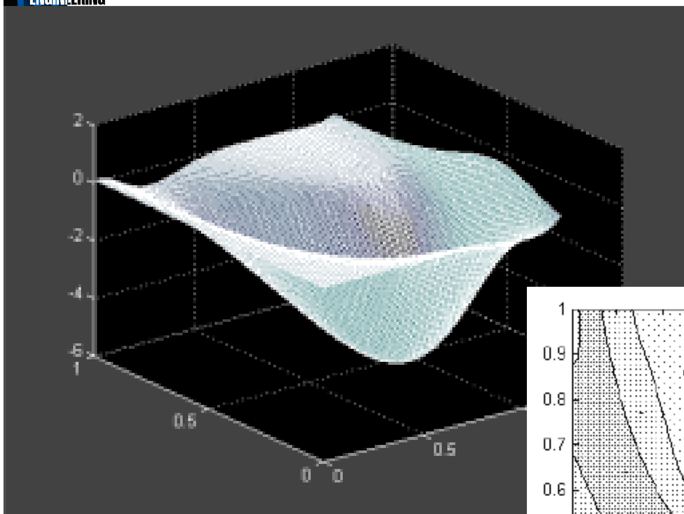
Representative Problem (Hughes, 1981)

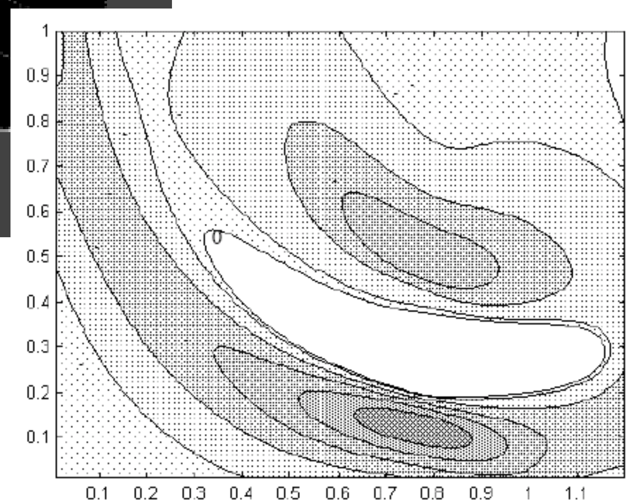
$Min\ \ f(x_1, x_2) = \alpha\ exp(-\beta)$
$u = x_1 - 0.8$
$v = x_2 - (a_1 + a_2\, u^2\, (1- u)^{1/2} - a_3\, u)$
$\alpha = -b_1 + b_2\, u^2\, (1+u)^{1/2} + b_3\, u$
$\beta = c_1\, v^2\, (1 - c_2\, v)/(1+ c_3\, u^2)$

a = [0.3, 0.6, 0.2]
b = [5, 26, 3]
c = [40, 1, 10]
x^* = [0.7395, 0.3144] $\quad f(x^*) = -5.0893$



Three Dimensional Surface and Curvature for Representative Test Problem

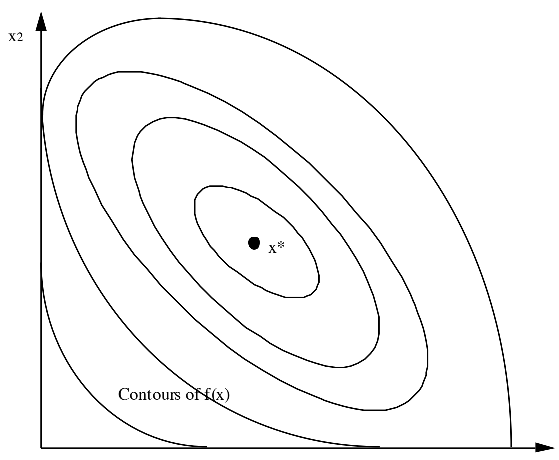




Regions where minimum eigenvalue is greater than:

[0, -10, -50, -100, -150, -200]

What conditions characterize an optimal solution?



Unconstrained Local Minimum
Necessary Conditions

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

$p^T \nabla^2 f(x^*)\, p \geq 0$ for $p \in \Re^n$

(positive semi-definite)

Unconstrained Local Minimum
Sufficient Conditions

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

$p^T \nabla^2 f(x^*)\, p > 0$ for $p \in \Re^n$

(positive definite)

For smooth functions, why are contours around optimum elliptical?
Taylor Series in n dimensions about x^*:

$$f(x) = f(x^*) + \nabla f(x^*)^T (x - x^*) + \frac{1}{2}(x - x^*)^T \nabla^2 f(x^*)(x - x^*) + O\left(\|x - x^*\|^3\right)$$

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^k

Take derivative wrt x, set LHS ≈ 0

$$0 \approx \nabla f(x) = \nabla f(x^k) + \nabla^2 f(x^k)\,(x - x^k) + O(||x - x^k||^2)$$
$$\Rightarrow \quad (x - x^k) \equiv d = -(\nabla^2 f(x^k))^{-1}\, \nabla f(x^k)$$

- $f(x)$ is convex (concave) if for all $x \in \Re^n$, $\nabla^2 f(x)$ is positive (negative) semidefinite i.e. $\min_j \lambda_j \geq 0$ ($\max_j \lambda_j \leq 0$)
- Method can fail if:
  - x^0 far from optimum
  - $\nabla^2 f$ is singular at any point
  - $f(x)$ is not smooth
- Search direction, d, requires solution of linear equations.
- Near solution:

$$\left\| x^{k+1} - x^* \right\| = O\left\| x^k - x^* \right\|^2$$

Basic Newton Algorithm - Line Search

0. Guess x^0, Evaluate $f(x^0)$.

1. At x^k, evaluate $\nabla f(x^k)$.

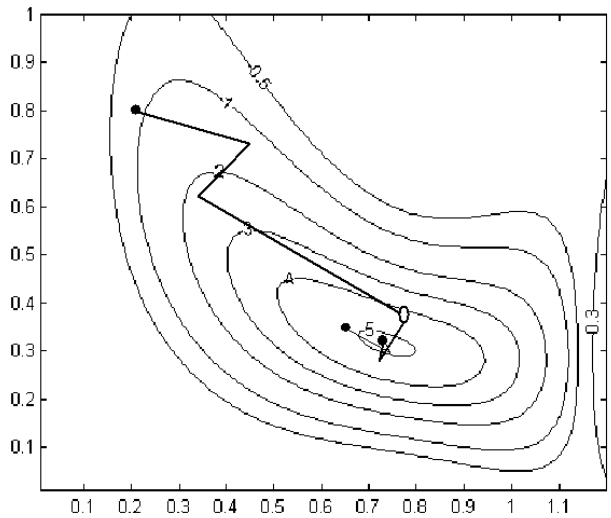
2. Evaluate $B^k = \nabla^2 f(x^k)$ or an approximation.

3. Solve: $\boldsymbol{B^k\, d = -\nabla f(x^k)}$
   If convergence error is less than tolerance:
   e.g., $||\nabla f(x^k)|| \leq \varepsilon$ and $||d|| \leq \varepsilon$ STOP, else go to 4.

4. Find α so that $0 < \alpha \leq 1$ and $f(x^k + \alpha\, d) < f(x^k)$ sufficiently (Each trial requires evaluation of $f(x)$)

5. $x^{k+1} = x^k + \alpha\, d$. Set $k = k + 1$ Go to 1.

Newton's Method - Convergence Path



Starting Points

[0.8, 0.2] needs steepest descent steps w/ line search up to 'O', takes 7 iterations to $\|\nabla f(x^*)\| \leq 10^{-6}$

[0.35, 0.65] converges in four iterations with full steps to $\|\nabla f(x^*)\| \leq 10^{-6}$

Newton's Method - Notes

- Choice of B^k determines method.
  - Steepest Descent: $B^k = \gamma I$
  - Newton: $B^k = \nabla^2 f(x)$
- With suitable B^k, performance may be good enough if $f(x^k + \alpha d)$ is sufficiently decreased (instead of minimized along line search direction).
- *Trust region extensions* to Newton's method provide very strong global convergence properties and very reliable algorithms.
- Local rate of convergence depends on choice of B^k.

$$\text{Newton – Quadratic Rate :} \quad \lim_{k\to\infty} \frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|^2} = K$$

$$\text{Steepest descent – Linear Rate :} \quad \lim_{k\to\infty} \frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|} < 1$$

$$\text{Desired? – Superlinear Rate :} \quad \lim_{k\to\infty} \frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|} = 0$$

Quasi-Newton Methods

Motivation:

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

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

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

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

$$B^{k+1} = B^k + \frac{(y - B^k s)y^T + y(y - B^k s)^T}{y^T s} - \frac{(y - B^k s)^T s\, y\, y^T}{(y^T s)(y^T s)}$$

$$(B^{k+1})^{-1} = H^{k+1} = H^k + \frac{ss^T}{s^T y} - \frac{H^k y\, y^T H^k}{y H^k y}$$

where: $s = x^{k+1} - x^k$

$y = \nabla f(x^{k+1}) - \nabla f(x^k)$

Quasi-Newton Methods

BFGS Formula (Broyden, Fletcher, Goldfarb, Shanno, 1970-71)

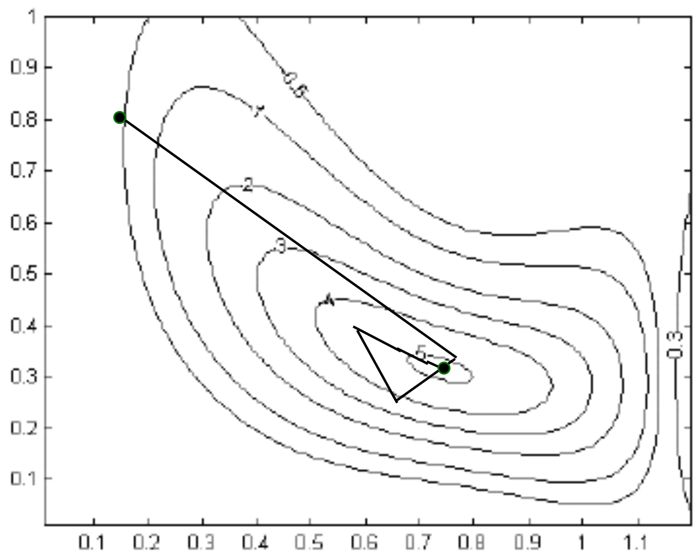
$$B^{k+1} = B^k + \frac{yy^T}{s^T y} - \frac{B^k s\, s^T B^k}{s B^k s}$$

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

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 α 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 if second derivatives are not available.

Quasi-Newton Method - BFGS Convergence Path



Starting Point

[0.2, 0.8] starting from $B^0 = I$, converges in 9 iterations to $||\nabla f(x^*)|| \leq 10^{-6}$

Sources For Unconstrained Software

Harwell (HSL)

IMSL

NAg - *Unconstrained Optimization Codes*

Netlib (www.netlib.org)

- •MINPACK
- •TOMS Algorithms, etc.

These sources contain various methods

- •Quasi-Newton
- •Gauss-Newton
- •Sparse Newton
- •Conjugate Gradient

Constrained Optimization (Nonlinear Programming)

Problem: $\quad Min_x \ f(x)$

$\quad s.t. \quad g(x) \le 0$

$\quad h(x) = 0$

where:

$f(x)$ - scalar objective function

x - n vector of variables

$g(x)$ - inequality constraints, m vector

$h(x)$ - meq equality constraints.

Sufficient Condition for Global 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?

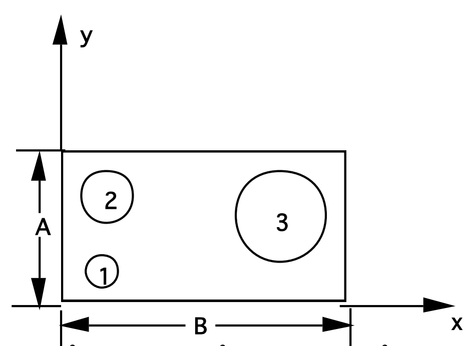
Variables: $A, B, (x_1, y_1), (x_2, y_2), (x_3, y_3)$

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.



$$\begin{cases} x_1, y_1 \ge R_1 & x_1 \le B - R_1, \ y_1 \le A - R_1 \\ x_2, y_2 \ge R_2 & x_2 \le B - R_2, \ y_2 \le A - R_2 \\ x_3, y_3 \ge R_3 & x_3 \le B - R_3, \ y_3 \le A - R_3 \end{cases}$$

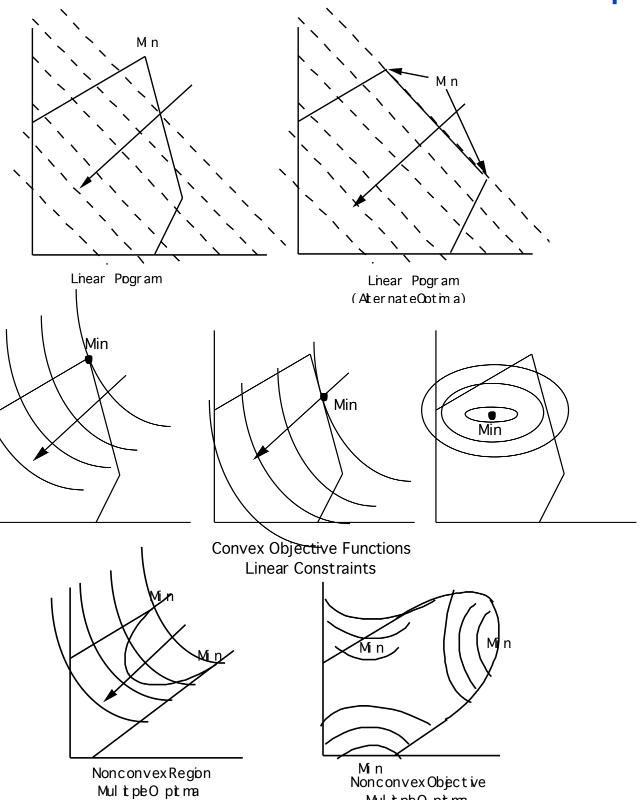
in box

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

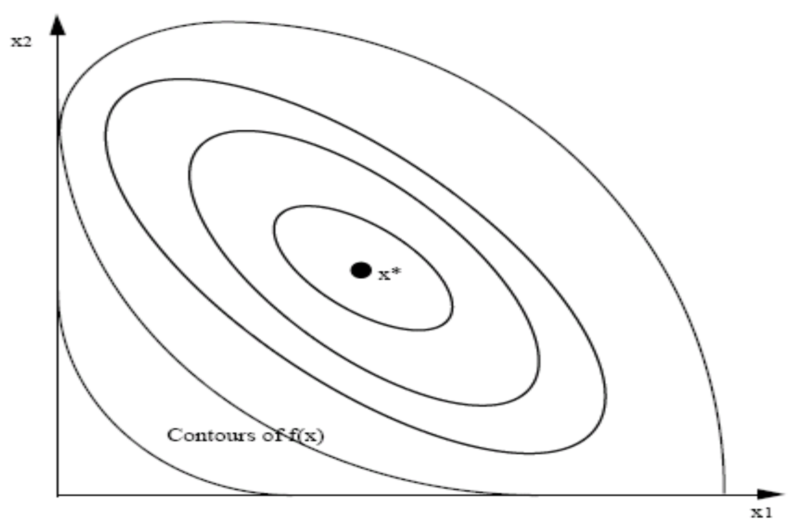
$$\begin{cases} (x_1 - x_2)^2 + (y_1 - y_2)^2 \ge (R_1 + R_2)^2 \\ (x_1 - x_3)^2 + (y_1 - y_3)^2 \ge (R_1 + R_3)^2 \\ (x_2 - x_3)^2 + (y_2 - y_3)^2 \ge (R_2 + R_3)^2 \end{cases}$$

no overlaps

Characterization of Constrained Optima



What conditions characterize an optimal solution?



<u>Unconstrained Local Minimum</u>
<u>Necessary Conditions</u>

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

$p^T \nabla^2 f(x^*)\, p \geq 0$ for $p \in \Re^n$

(positive semi-definite)

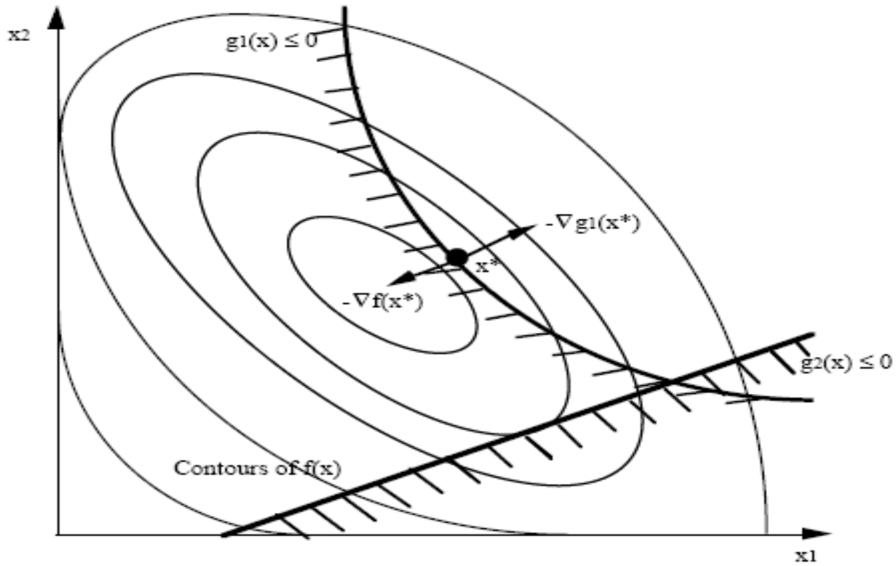
<u>Unconstrained Local Minimum</u>
<u>Sufficient Conditions</u>

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

$p^T \nabla^2 f(x^*)\, p > 0$ for $p \in \Re^n$

(positive definite)

Optimal solution for inequality constrained problem



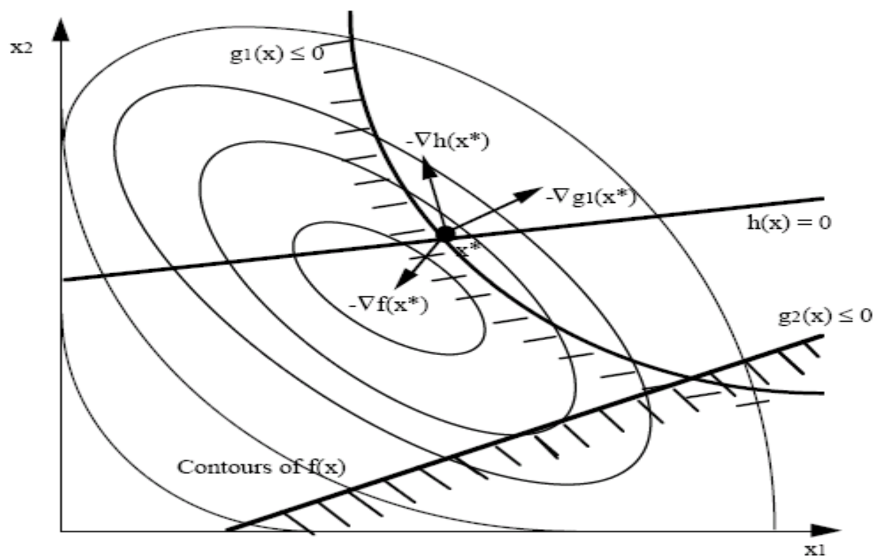
Min f(x)

s.t. $g(x) \le 0$

Analogy: Ball rolling down valley pinned by fence

Note: Balance of forces (∇f, ∇g_1)

Optimal solution for general constrained problem



Problem: Min f(x)

s.t. $g(x) \le 0$

$h(x) = 0$

Analogy: Ball rolling on rail pinned by fences

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

Optimality conditions for local optimum

Necessary First Order Karush Kuhn - Tucker Conditions

$\nabla L(x^*, u, v) = \nabla f(x^*) + \nabla g(x^*)\, u + \nabla h(x^*)\, v = 0$
(Balance of Forces)
$u \geq 0$ (Inequalities act in only one direction)
$g(x^*) \leq 0,\ h(x^*) = 0$ (Feasibility)
$u_j\, g_j(x^*) = 0$ (Complementarity: either $g_j(x^*) = 0$ or $u_j = 0$)
u, v are "weights" for "forces," known as KKT multipliers, shadow prices, dual variables

"To guarantee that a local NLP solution satisfies KKT conditions, a constraint qualification is required. E.g., the *Linear Independence Constraint Qualification* (LICQ) requires active constraint gradients, $[\nabla g_A(x^*)\ \ \nabla h(x^*)]$, to be linearly independent. Also, under LICQ, KKT multipliers are uniquely determined."

Necessary (Sufficient) Second Order Conditions

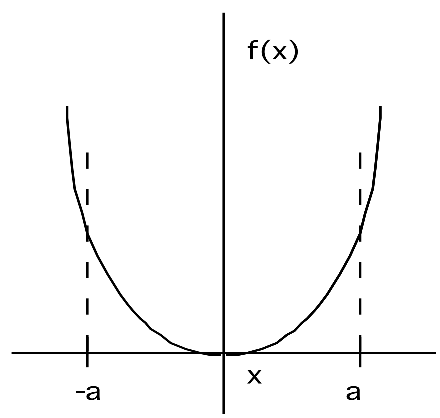
- Positive curvature in "constraint" directions.
- $p^T \nabla^2 L(x^*)\, p \geq 0$ $(p^T \nabla^2 L(x^*)\, p > 0)$
  where p are the constrained directions: $\nabla h(x^*)^T p = 0$
  for $g_i(x^*)=0$, $\nabla g_i(x^*)^T p = 0$, for $u_i > 0$, $\nabla g_i(x^*)^T p \leq 0$, for $u_i = 0$

Single Variable Example of KKT Conditions

Min $(x)^2$ s.t. $-a \leq x \leq a,\ a > 0$
$x^* = 0$ is seen by inspection

Lagrange function :
$L(x, u) = x^2 + u_1(x-a) + u_2(-a-x)$

First Order KKT conditions:
$\nabla L(x, u) = 2x + u_1 - u_2 = 0$
$u_1\,(x-a) = 0$
$u_2\,(-a-x) = 0$
$-a \leq x \leq a$ $\qquad u_1, u_2 \geq 0$



Consider three cases:

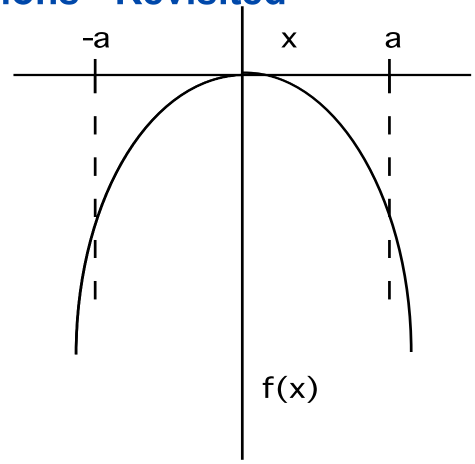
- $u_1 \geq 0,\ u_2 = 0$ Upper bound is active, $x = a,\ u_1 = -2a,\ u_2 = 0$
- $u_1 = 0,\ u_2 \geq 0$ Lower bound is active, $x = -a,\ u_2 = -2a,\ u_1 = 0$
- $u_1 = u_2 = 0$ Neither bound is active, $u_1 = 0,\ u_2 = 0,\ x = 0$

Second order conditions $(x^*, u_1, u_2 = 0)$

$$\nabla_{xx} L(x^*, u^*) = 2$$
$$p^T \nabla_{xx} L(x^*, u^*)\, p = 2\,(\Delta x)^2 > 0$$

Single Variable Example of KKT Conditions - Revisited

Min $-(x)^2$ s.t. $-a \le x \le a,\ a > 0$
$x^* = \pm a$ is seen by inspection



Lagrange function :
$L(x, u) = -x^2 + u_1(x-a) + u_2(-a-x)$

First Order KKT conditions:
$\nabla L(x, u) = -2x + u_1 - u_2 = 0$
$u_1 (x-a) = 0$
$u_2 (-a-x) = 0$
$-a \le x \le a \qquad u_1, u_2 \ge 0$

Consider three cases:

- $u_1 \ge 0,\ u_2 = 0$ — Upper bound is active, $x = a,\ u_1 = 2a,\ u_2 = 0$
- $u_1 = 0,\ u_2 \ge 0$ — Lower bound is active, $x = -a,\ u_2 = 2a,\ u_1 = 0$
- $u_1 = u_2 = 0$ — Neither bound is active, $u_1 = 0,\ u_2 = 0,\ x = 0$

Second order conditions $(x^*, u_1, u_2 = 0)$

$$\nabla_{xx} L(x^*, u^*) = -2$$
$$p^T \nabla_{xx} L(x^*, u^*)\, p = -2(\Delta x)^2 < 0$$

Interpretation of Second Order Conditions

For $x = a$ or $x = -a$, we require the allowable direction to satisfy the active constraints exactly. Here, any point along the allowable direction, x^* must remain at its bound.

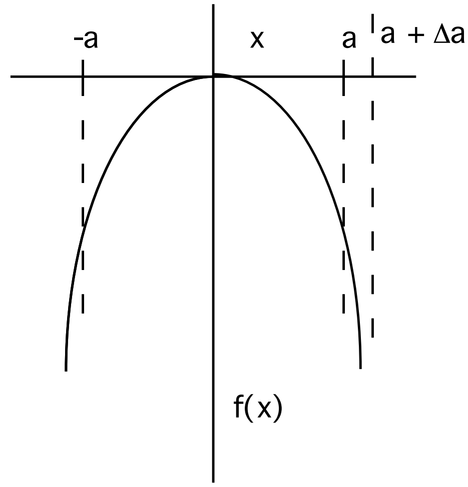
For this problem, however, there are no nonzero allowable directions that satisfy this condition. Consequently the solution x* is defined entirely by the active constraint. The condition:

$$p^T \nabla_{xx} L(x^*, u^*, v^*)\, p > 0$$

for the allowable directions, is *vacuously* satisfied - because there are *no* allowable directions that satisfy $\nabla g_A(x^*)^T p = 0$. Hence, *sufficient* second order conditions are satisfied.

As we will see, sufficient second order conditions are satisfied by linear programs as well.

Role of KKT Multipliers



Also known as:

- Shadow Prices
- Dual Variables
- Lagrange Multipliers

Suppose a in the constraint is increased to $a + \Delta a$

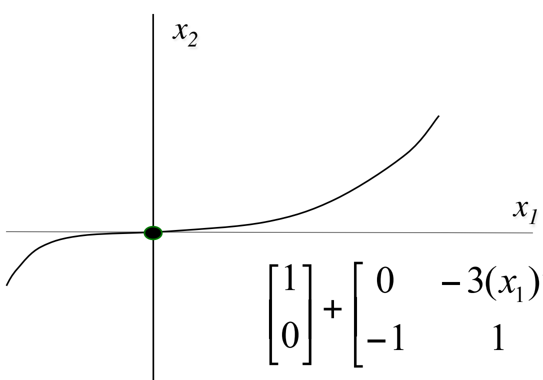
$$f(x^*) = -(a + \Delta a)^2$$

and

$$[f(x^*, a + \Delta a) - f(x^*, a)]/\Delta a = -2a - \Delta a$$

$$df(x^*)/da = -2a = -u_1$$

Another Example: Constraint Qualifications



$$\begin{aligned} &Min \quad x_1 \\ &s.t. \ x_2 \geq 0 \\ &\qquad x_2 \leq (x_1)^3 \\ &\qquad x_1^* = x_2^* = 0 \end{aligned}$$

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & -3(x_1)^2 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \neq 0$$

$$-x_2 \leq 0, u_1 \geq 0, u_1 x_2 = 0$$

$$x_2 - (x_1)^3 \leq 0, u_2 \geq 0, u_2(x_2 - (x_1)^3) = 0$$

KKT conditions not satisfied at NLP solution

Because a CQ is not satisfied (e.g., LICQ)

Special Cases of Nonlinear Programming

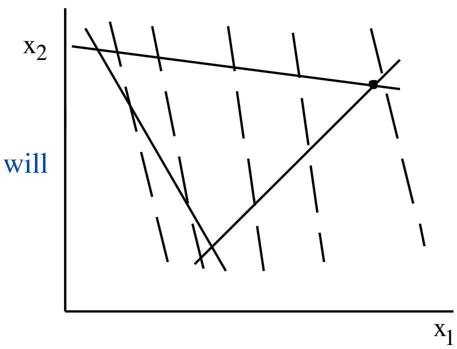
Linear Programming:

Min $c^T x$

s.t. $Ax \le b$

$Cx = d,\ x \ge 0$

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.

Linear Programming Example

Simplex Method

$$\begin{aligned} &\text{Min } -2x_1 - 3x_2 \\ &\text{s.t. } 2x_1 + x_2 \le 5 \\ &\qquad x_1, x_2 \ge 0 \end{aligned} \quad \Rightarrow \quad \begin{aligned} &\text{Min } -2x_1 - 3x_2 \\ &\text{s.t. } 2x_1 + x_2 + x_3 = 5 \\ &\qquad x_1, x_2, x_3 \ge 0 \\ &\qquad \text{(add slack variable)} \end{aligned}$$

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 $x_3 \ge 0$

x_1	x_2	x_3	f	b
2	1	1	0	5
$\underline{-4}$	0	$\underline{-3}$	1	-15

f can no longer be decreased! ***Optimal***

Underlined terms are -(reduced gradients); nonbasic variables (x_1, x_3), basic variable x_2

Quadratic Programming

Problem: Min $a^T x + 1/2\, x^T B x$

$A x \le b$

$C x = d$

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

   $\Rightarrow$ Min $\Sigma_j (z_{j+} + z_{j-})$

   s.t. $a + Bx + A^T u + C^T v = z_+ - z_-$

   $Ax - b + s = 0$

   $Cx - d = 0$

   $u, s, z_+, z_- \ \ge\ 0$

   $\{u_j\, s_j = 0\}$

   with complicating conditions.

2) If B is positive definite, QP solution is unique.
   If B is pos. semidefinite, optimum value is unique.

3) Other methods for solving QP's (faster)
   - Complementary Pivoting (Lemke)
   - Range, Null Space methods (Gill, Murray).

Portfolio Planning Problem

Definitions:

x_i - fraction or amount invested in security i

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

μ_i - average r(t) over T years, i.e.

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

$$Max \sum_i \mu_i x_i$$

$$s.t. \sum_i x_i = 1$$

$$x_i \ge 0,\ etc.$$

Note: maximize average return, no accounting for risk.

Portfolio Planning Problem

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

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

$$\begin{aligned} & Min \quad x^T S x \\ & s.t. \quad \sum_i x_i = 1 \\ & \sum_i \mu_i x_i \geq R \\ & x_i \geq 0, \; etc. \end{aligned}$$

Example: 3 investments

		μ_j
1.	IBM	1.3
2.	GM	1.2
3.	Gold	1.08

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

Portfolio Planning Problem - GAMS

```
SIMPLE PORTFOLIO INVESTMENT PROBLEM (MARKOWITZ)

OPTION LIMROW=0;
OPTION LIMXOL=0;

VARIABLES IBM, GM, GOLD, OBJQP, OBJLP;

EQUATIONS E1,E2,QP,LP;

LP.. OBJLP =E= 1.3*IBM + 1.2*GM + 1.08*GOLD;

QP.. OBJQP =E= 3*IBM**2 + 2*IBM*GM - IBM*GOLD
+    2*GM**2 - 0.8*GM*GOLD + GOLD**2;

E1..1.3*IBM + 1.2*GM + 1.08*GOLD =G= 1.15;

E2.. IBM + GM + GOLD =E= 1;

IBM.LO = 0.;
IBM.UP = 0.75;
GM.LO = 0.;
GM.UP = 0.75;
GOLD.LO = 0.;
GOLD.UP = 0.75;

MODEL PORTQP/QP,E1,E2/;

MODEL PORTLP/LP,E2/;

SOLVE PORTLP USING LP MAXIMIZING OBJLP;

     SOLVE PORTQP USING NLP MINIMIZING OBJQP;
```

Portfolio Planning Problem - GAMS

```
S O L V E   S U M M A R Y
**** 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.

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
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
DERIVITIVE POOL            8        CONSTANT POOL                 3
CODE LENGTH               95
GENERATION TIME            =      2.360 SECONDS
EXECUTION TIME       =     3.510 SECONDS
```


Portfolio Planning Problem - GAMS

```
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
M I N O S      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.

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
SIMPLE PORTFOLIO INVESTMENT PROBLEM (MARKOWITZ)
Model Statistics     SOLVE PORTQP USING NLP FROM LINE 34
EXECUTION TIME          =       1.090  SECONDS
```

Algorithms for Constrained Problems

Motivation: Build on unconstrained methods wherever possible.

Classification of Methods:

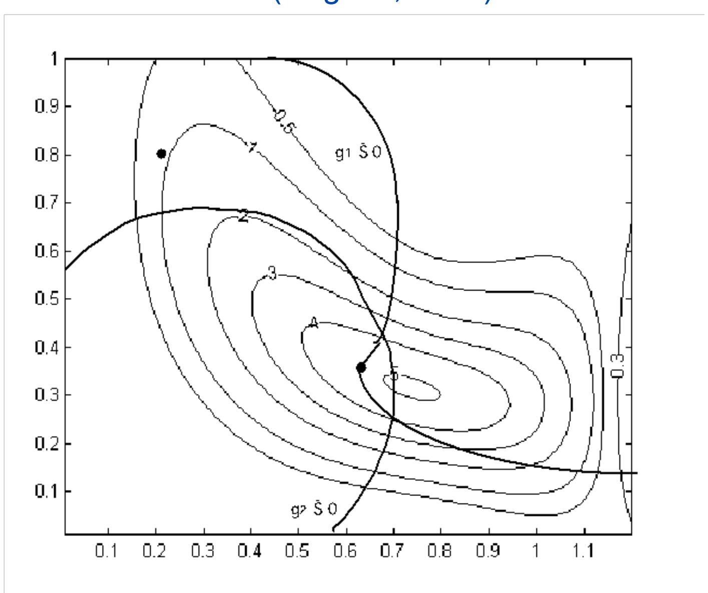
- Reduced Gradient Methods - (with Restoration) GRG2, CONOPT
- Reduced Gradient Methods - (without Restoration) MINOS
- Successive Quadratic Programming - generic implementations
- Penalty Functions - popular in 1970s, but fell into disfavor. Barrier Methods have been developed recently and are again popular.
- Successive Linear Programming - only useful for "mostly linear" problems

We will concentrate on algorithms for first four classes.

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

Representative Constrained Problem

(Hughes, 1981)



Min $f(x_1, x_2) = \alpha \exp(-\beta)$

$g_1 = (x_2+0.1)^2[x_1^2+2(1-x_2)(1-2x_2)] - 0.16 \le 0$

$g_2 = (x_1 - 0.3)^2 + (x_2 - 0.3)^2 - 0.16 \le 0$

$x^* = [0.6335, 0.3465] \qquad f(x^*) = -4.8380$

Reduced Gradient Method with Restoration

(GRG2/CONOPT)

$$\begin{array}{ll} \text{Min} & f(x) \\ \text{s.t.} & g(x) + s = 0 \text{ (add slack variable)} \\ & h(x) = 0 \\ & a \le x \le b,\ s \ge 0 \end{array} \quad \Rightarrow \quad \begin{array}{ll} \text{Min} & f(z) \\ & \text{s.t. } c(z) = 0 \\ & a \le z \le b \end{array}$$

Partition variables into:

- z_B - dependent or <u>basic</u> variables
- z_N - <u>nonbasic</u> variables, fixed at a bound
- z_S - independent or superbasic variables

Modified KKT Conditions

$$\nabla f(z) + \nabla c(z)\lambda - \nu_L + \nu_U = 0$$

$$c(z) = 0$$

$$z^{(i)} = z_U^{(i)} \quad or \quad z^{(i)} = z_L^{(i)}, \quad i \in N$$

$$\nu_U^{(i)},\ \nu_L^{(i)} = 0, \quad i \notin N$$

Reduced Gradient Method with Restoration

(GRG2/CONOPT)

$$\begin{array}{ll} a) & \nabla_S f(z) + \nabla_S c(z)\lambda = 0 \\ b) & \nabla_B f(z) + \nabla_B c(z)\lambda = 0 \\ c) & \nabla_N f(z) + \nabla_N c(z)\lambda - \nu_L + \nu_U = 0 \\ d) & z^{(i)} = z_U^{(i)} \quad or \quad z^{(i)} = z_L^{(i)}, \quad i \in N \\ e) & c(z) = 0 \Rightarrow z_B = z_B(z_S) \end{array}$$

- Solve bound constrained problem in space of superbasic variables (apply gradient projection algorithm)
- Solve (e) to eliminate z_B
- Use (a) and (b) to calculate *reduced gradient* wrt z_S.
- Nonbasic variables z_N (temporarily) fixed (d)
- Repartition based on signs of ν, if z_s remain at bounds or if z_B violate bounds

Definition of Reduced Gradient

$$\frac{df}{dz_S} = \frac{\partial f}{\partial z_S} + \frac{dz_B}{dz_S}\frac{\partial f}{\partial z_B}$$

Because $c(z) = 0$, we have :

$$dc = \left[\frac{\partial c}{\partial z_S}\right]^T dz_S + \left[\frac{\partial c}{\partial z_B}\right]^T dz_B = 0$$

$$\frac{dz_B}{dz_S} = -\left[\frac{\partial c}{\partial z_S}\right]\left[\frac{\partial c}{\partial z_B}\right]^{-1} = -\nabla_{z_S} c\left[\nabla_{z_B} c\right]^{-1}$$

This leads to :

$$\frac{df}{dz_S} = \nabla_S f(z) - \nabla_S c\left[\nabla_B c\right]^{-1}\nabla_B f(z) = \nabla_S f(z) + \nabla_S c(z)\lambda$$

- By remaining feasible always, *$c(z) = 0$, $a \le z \le b$,* one can apply an unconstrained algorithm (quasi-Newton) using *(df/dz_S), using (b)*
- Solve problem in reduced space of z_S variables, using (e).

Example of Reduced Gradient

$$\begin{aligned} &Min \quad x_1^2 - 2x_2 \\ &s.t. \quad 3x_1 + 4x_2 = 24 \\ &\nabla c^T = [3 \;\; 4],\; \nabla f^T = [2x_1 \;\; -2] \end{aligned}$$

Let $z_S = x_1,\; z_B = x_2$

$$\frac{df}{dz_S} = \frac{\partial f}{\partial z_S} - \nabla_{z_S} c\left[\nabla_{z_B} c\right]^{-1}\frac{\partial f}{\partial z_B}$$

$$\frac{df}{dx_1} = 2x_1 - 3\left[4\right]^{-1}\left(-2\right) = 2x_1 + 3/2$$

If ∇c^T is (m x n); $\nabla z_S c^T$ is m x (n-m); $\nabla z_B c^T$ is (m x m)

(df/dz_S) is the change in *f* along constraint direction per unit change in z_S

Gradient Projection Method
(superbasic → nonbasic variable partition)

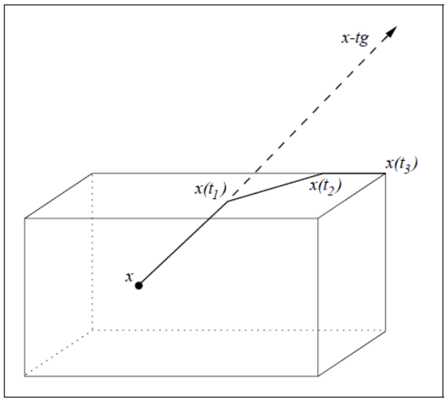


Figure 16.5 The piecewise linear path $x(t)$, for an example in $\mathbf{R}^3$.

Define the projection of an arbitrary point x onto box feasible region.

The ith component is given by

$$P(x, l, u)_i = \begin{cases} l_i & \text{if} \quad x_i < l_i, \\ x_i & \text{if} \quad x_i \in [l_i, u_i], \\ u_i & \text{if} \quad x_i > u_i. \end{cases}$$

Piecewise linear path $x(t)$ starting at the reference point x_0 and obtained by projecting steepest descent (or any search) direction at x_0 onto the box region is given by

$$x(t) = P(x^0 - tg, l, u),$$

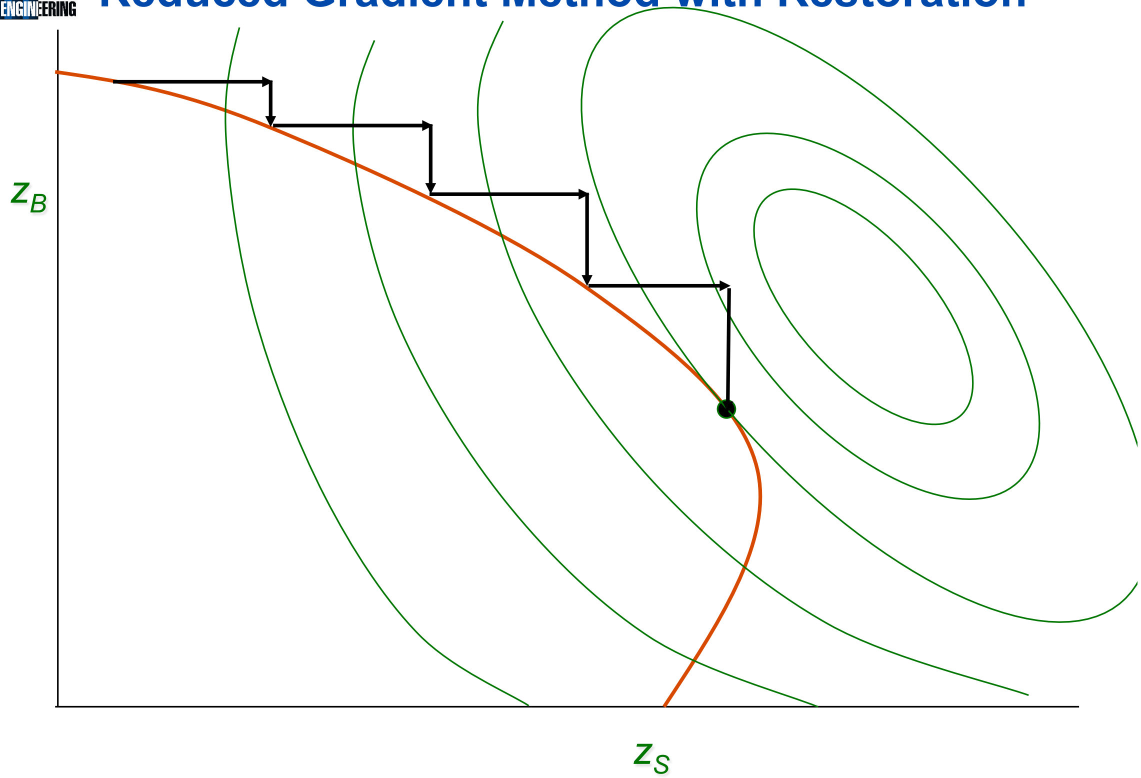
where g is the reduced gradient, t is the stepsize.

Also, can adapt to (quasi-) Newton method.

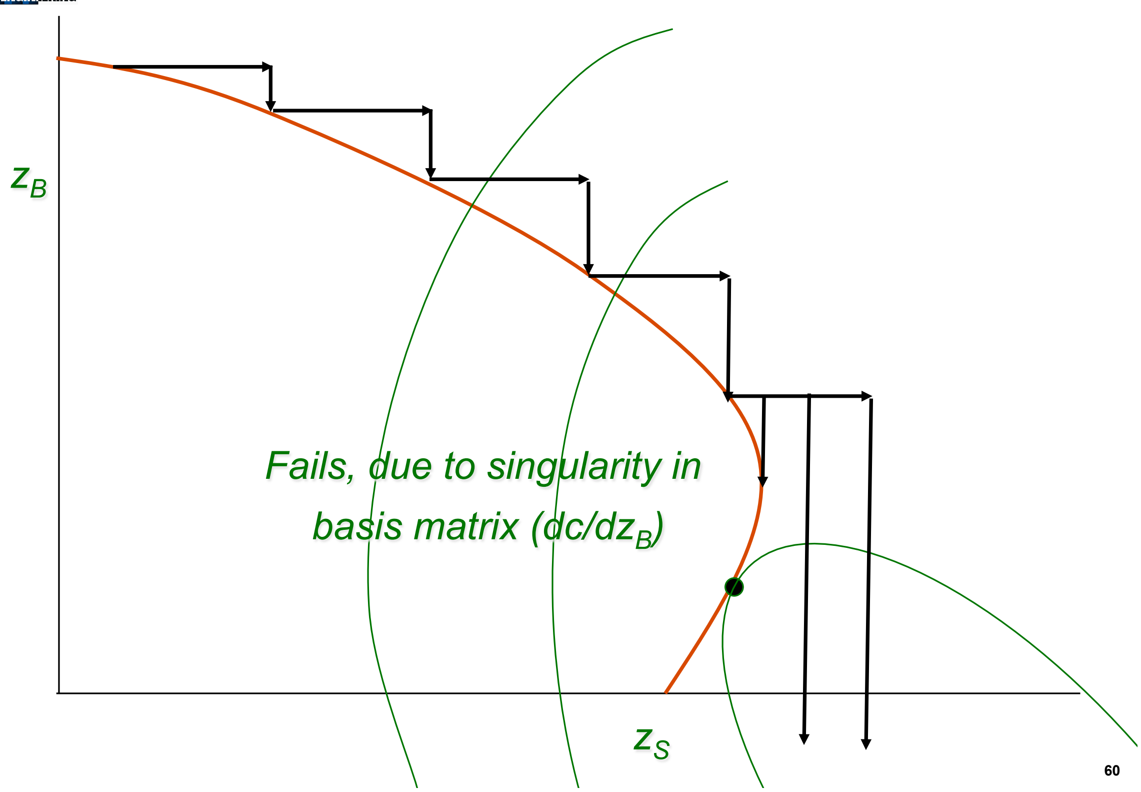
Sketch of GRG Algorithm

1. Initialize problem and obtain a feasible point at z^0
2. At feasible point z^k, partition variables z into z_N, z_B, z_S
3. Calculate reduced gradient, (df/dz_S)
4. Evaluate search direction for z_S, $d = B^{-1}(df/dz_S)$
5. Perform a line search.
   - Find $\alpha \in (0,1]$ with $z_S := z_S^k + \alpha\, d$
   - Solve for $c(z_S^k + \alpha\, d, z_B, z_N) = 0$
   - If $f(z_S^k + \alpha\, d, z_B, z_N) < f(z_S^k, z_B, z_N)$, set $z_S^{k+1} = z_S^k + \alpha\, d$, $k := k+1$
6. If $||(df/dz_S)|| < \varepsilon$, Stop. Else, go to 2.

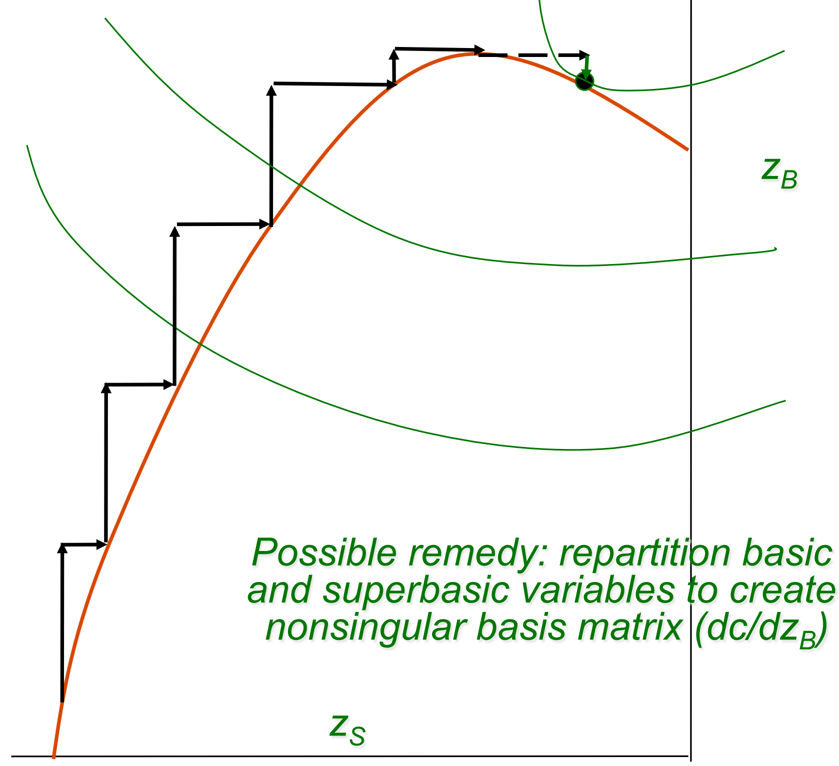
Reduced Gradient Method with Restoration



Reduced Gradient Method with Restoration



Reduced Gradient Method with Restoration



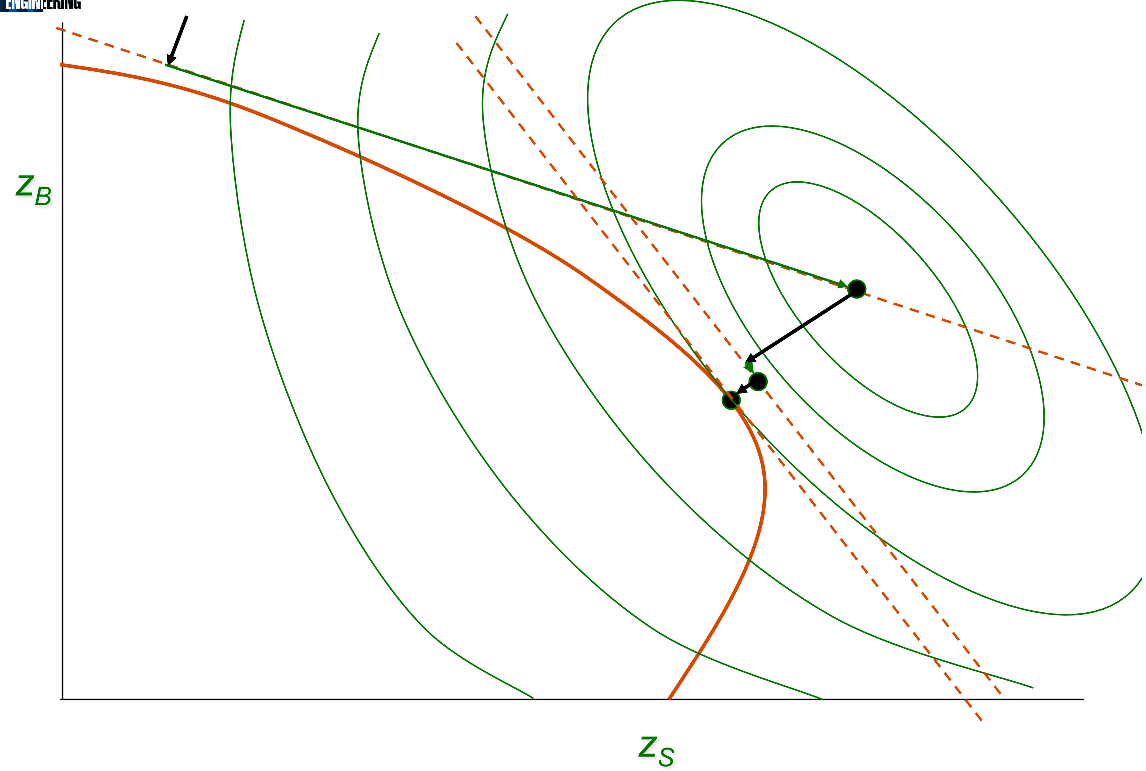
GRG Algorithm Properties

1. 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. CONOPT is implemented in GAMS, AIMMS and AMPL
3. GRG2 uses Q-N for small problems but can switch to conjugate gradients if problem gets large. CONOPT uses exact second derivatives.
4. Convergence of $c(z_S, z_B, z_N) = 0$ can get <u>very</u> expensive because $\nabla c(z)$ is calculated repeatedly.
5. Safeguards can be added so that restoration (step 5.) can be dropped and efficiency increases.

<u>Representative Constrained Problem</u> Starting Point [0.8, 0.2]

- GINO Results - 14 iterations to $\|\nabla f(x^*)\| \leq 10^{-6}$
- CONOPT Results - 7 iterations to $\|\nabla f(x^*)\| \leq 10^{-6}$ from feasible point.

Reduced Gradient Method without Restoration



Reduced Gradient Method without Restoration (MINOS/Augmented)

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

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

Extend to nonlinear problems, through successive linearization

Develop major iterations (linearizations) and minor iterations (GRG solutions) .

Strategy: (Robinson, Murtagh & Saunders)

1. Partition variables into basic, nonbasic variables and superbasic variables..
2. Linearize active constraints at z^k
   $$D^k z = r^k$$
3. Let $\psi = f(z) + \lambda^T c(z) + \beta\,(c(z)^T c(z))$ (Augmented Lagrange),
4. Solve linearly constrained problem:
   $$\begin{aligned} Min \quad & \psi(z) \\ s.t. \quad & Dz = r \\ & a \le z \le b \end{aligned}$$
   using reduced gradients to get z^{k+1}
5. Set $k=k+1$, go to 2.
6. Algorithm terminates when no movement between steps 2) and 4).

MINOS/Augmented 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 routines.
2. No restoration takes place, nonlinear constraints are reflected in $\psi(z)$ during step 3). MINOS is more efficient than GRG.
3. Major iterations (steps 3) - 4)) converge at a quadratic rate.
4. Reduced gradient methods are complicated, monolithic codes: hard to integrate efficiently into modeling software.

Representative Constrained Problem – Starting Point [0.8, 0.2]
MINOS Results: 4 major iterations, 11 function calls
to $\|\nabla f(x^*)\| \leq 10^{-6}$

Successive Quadratic Programming (SQP)

Motivation:

- Take KKT conditions, expand in Taylor series about current point.
- Take Newton step (QP) to determine next point.

Derivation – KKT Conditions

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

$h(x^*) = 0$

$g_A(x^*) = 0,$ where g_A are the active constraints.

Newton - Step

$$\begin{bmatrix} \nabla_{xx} L & \nabla g_A & \nabla h \\ \nabla g_A^T & 0 & 0 \\ \nabla h^T & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta u \\ \Delta v \end{bmatrix} = - \begin{bmatrix} \nabla_x L(x^k, u^k, v^k) \\ g_A(x^k) \\ h(x^k) \end{bmatrix}$$

Requirements:

- $\nabla_{xx} L$ must be calculated and should be 'regular'
- correct active set g_A
- good estimates of u^k, v^k

SQP Chronology

1. *Wilson (1963)*
 - active set can be determined by solving QP:

$$\min_{d} \quad \nabla f(x_k)^T d + 1/2\, d^T\, \nabla_{xx} L(x_k, u_k, v_k)\, d$$
$$\text{s.t.} \quad g(x_k) + \nabla g(x_k)^T d \le 0$$
$$h(x_k) + \nabla h(x_k)^T d = 0$$

2. *Han (1976), (1977), Powell (1977), (1978)*
 - approximate $\nabla_{xx}L$ using a positive definite quasi-Newton update (BFGS)
 - 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).

Elements of SQP – Hessian Approximation

What about $\nabla_{xx}L$?

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

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

$$B^{k+1} = B^k + \frac{yy^T}{s^T y} - \frac{B^k s\; s^T B^k}{s\, B^k s}$$

BFGS Formula $\quad s = x^{k+1} - x^k$

$$y = \nabla L(x^{k+1}, u^{k+1}, v^{k+1}) - \nabla L(x^k, u^{k+1}, v^{k+1})$$

- second derivatives approximated by change in gradients
- positive definite B^k ensures *unique* QP solution

Elements of SQP – Search Directions

How do we obtain search directions?

- Form QP and let QP determine constraint activity
- At each iteration, k, solve:

$$\min_{d} \quad \nabla f(x^k)^T d + 1/2\, d^T B^k d$$

$$\text{s.t.} \quad g(x^k) + \nabla g(x^k)^T d \le 0$$

$$h(x^k) + \nabla h(x^k)^T d = 0$$

Convergence from poor starting points

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

Exact Penalty Function

$$\psi(x) = f(x) + \mu\, [\Sigma \max(0, g_j(x)) + \Sigma\, |h_j(x)|]$$

$$\mu > \max_j \{| u_j |, | v_j |\}$$

Augmented Lagrange Function

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

Newton-Like Properties for SQP

Fast Local Convergence

$B = \nabla_{xx} L$	Quadratic
$\nabla_{xx} L$ is p.d and B is Q-N	1 step Superlinear
B is Q-N update, $\nabla_{xx} L$ not p.d	2 step Superlinear

Enforce Global Convergence

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

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

Basic SQP Algorithm

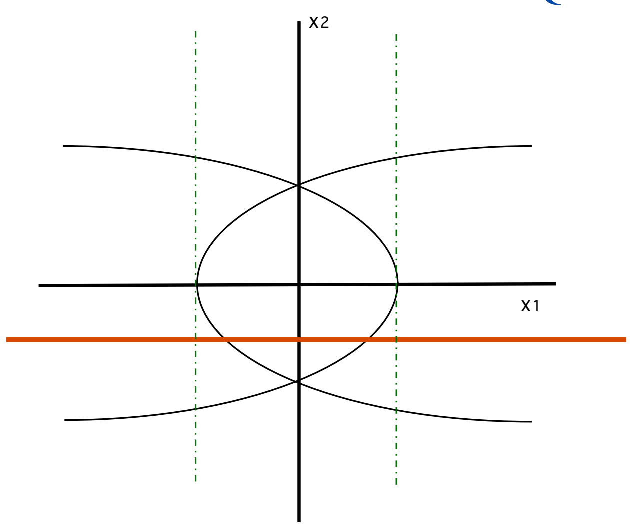
0. Guess x^0, Set $B^0 = I$ (Identity). Evaluate $f(x^0)$, $g(x^0)$ and $h(x^0)$.
1. At x^k, evaluate $\nabla f(x^k)$, $\nabla g(x^k)$, $\nabla h(x^k)$.
2. If k > 0, update B^k using the BFGS Formula.
3. Solve: $\quad Min_d \;\; \nabla f(x^k)^T d + 1/2\, d^T B^k d$

$$
\begin{aligned}
s.t. \quad & g(x^k) + \nabla g(x^k)^T d \le 0 \\
& h(x^k) + \nabla h(x^k)^T d = 0
\end{aligned}
$$

   If KKT error less than tolerance: $||\nabla L(x^*)|| \le \varepsilon$, $||h(x^*)|| \le \varepsilon$, $||g(x^*)_+|| \le \varepsilon$. STOP, else go to 4.
4. Find α so that $0 < \alpha \le 1$ and $\psi(x^k + \alpha d) < \psi(x^k)$ sufficiently (Each trial requires evaluation of $f(x)$, $g(x)$ and $h(x)$).
5. $x^{k+1} = x^k + \alpha\, d$. Set k = k + 1 Go to 2.

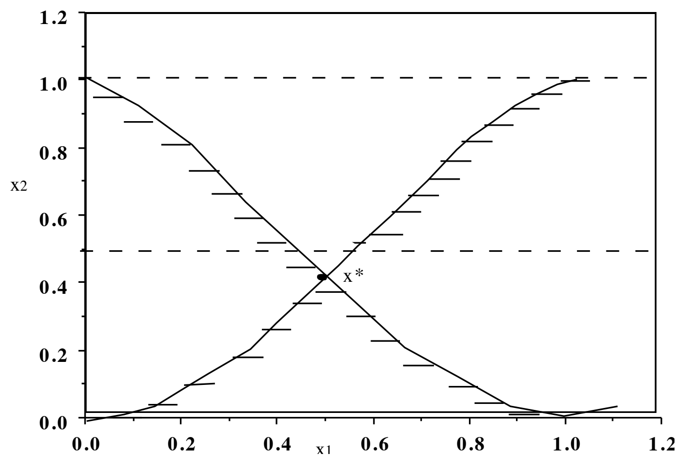
Problems with SQP

Nonsmooth Functions - Reformulate
Ill-conditioning - Proper scaling
Poor Starting Points – Trust Regions can help
Inconsistent Constraint Linearizations
- Can lead to infeasible QP's



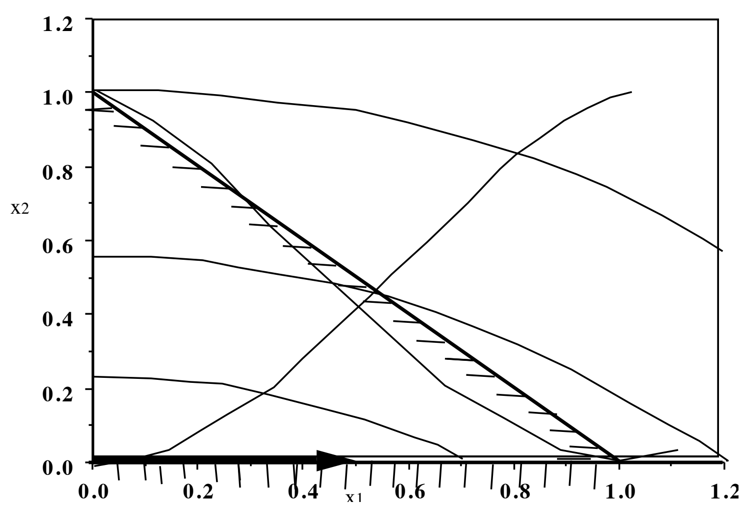
$$
\begin{aligned}
Min \quad & x_2 \\
s.t. \quad & 1 + x_1 - (x_2)^2 \le 0 \\
& 1 - x_1 - (x_2)^2 \le 0 \\
& x_2 \ge -1/2
\end{aligned}
$$

SQP Test Problem



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

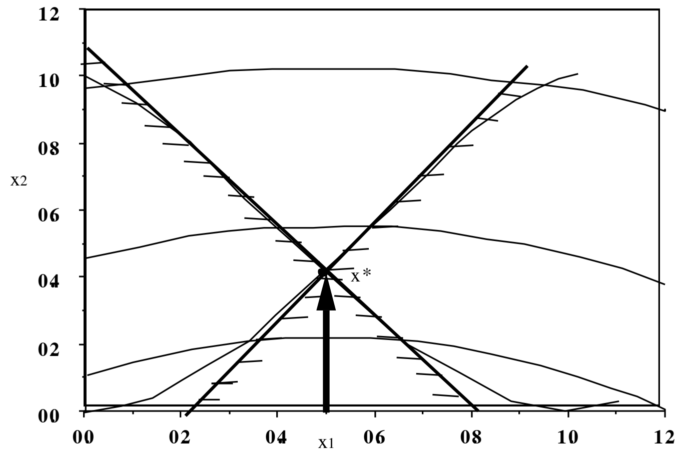
SQP Test Problem – First Iteration



Start from the origin ($x_0 = [0, 0]^T$) with $B^0 = I$, form:

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

SQP Test Problem – Second Iteration



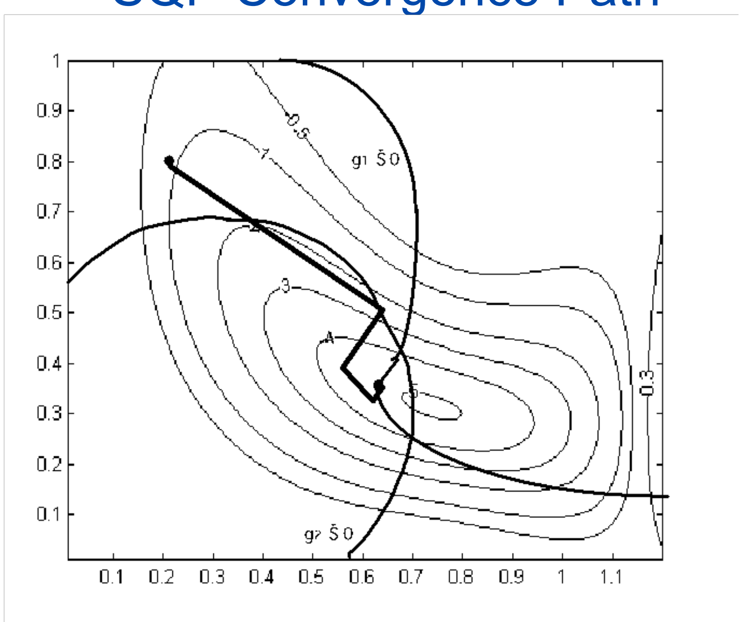
From $x_1 = [0.5, 0]^T$ with $B^1 = I$
(no update from BFGS possible), form:

$$\begin{aligned} &Min \quad d_2 + 1/2\,(d_1^2 + d_2^2) \\ &s.t. \quad -1.25\, d_1 - d_2 + 0.375 \leq 0 \\ &\qquad\;\; 1.25\, d_1 - d_2 + 0.375 \leq 0 \end{aligned}$$

$d = [0, 0.375]^T$ with $\mu_1 = 0.5$ and $\mu_2 = 0.5$

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

Representative Constrained Problem SQP Convergence Path



Starting Point [0.8, 0.2] - starting from $B^0 = I$ and staying in bounds and linearized constraints; converges in 8 iterations to $||\nabla f(x^*)|| \leq 10^{-6}$

Barrier Methods for Large-Scale Nonlinear Programming

Original Formulation

$$\min_{x \in \Re^n} f(x)$$
$$\text{s.t} \quad c(x) = 0$$
$$x \geq 0$$

Can generalize for $a \leq x \leq b$

Barrier Approach

$$\min_{x \in \Re^n} \varphi_\mu(x) = f(x) - \mu \sum_{i=1}^{n} \ln x_i$$
$$\text{s.t} \quad c(x) = 0$$

⇒As $\mu \rightarrow 0$, $x^*(\mu) \rightarrow x^*$ Fiacco and McCormick (1968)

Solution of the Barrier Problem

⇒**Newton Directions (KKT System)**

$$\begin{aligned} \nabla f(x) + A(x)\lambda - v &= 0 \\ Xv - \mu e &= 0 \\ c(x) &= 0 \end{aligned}$$

$e^T = [1,1,1\ldots]$, $X = diag(x)$

$A = \nabla c(x)$, $W = \nabla_{xx} L(x,\lambda,\nu)$

⇒ **Reducing the System**

$$d_v = \mu X^{-1} e - v - X^{-1} V d_x$$

$$\begin{bmatrix} W + \Sigma & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} d_x \\ \lambda^+ \end{bmatrix} = - \begin{bmatrix} \nabla \varphi_\mu \\ c \end{bmatrix} \qquad \Sigma = X^{-1} V$$

IPOPT Code – www.coin-or.org

Global Convergence of Newton-based Barrier Solvers

Merit Function

Exact Penalty: $P(x, \eta) = f(x) + \eta \,||c(x)||$

Aug'd Lagrangian: $L^*(x, \lambda, \eta) = f(x) + \lambda^T c(x) + \eta \,||c(x)||^2$

Assess Search Direction (e.g., from IPOPT)

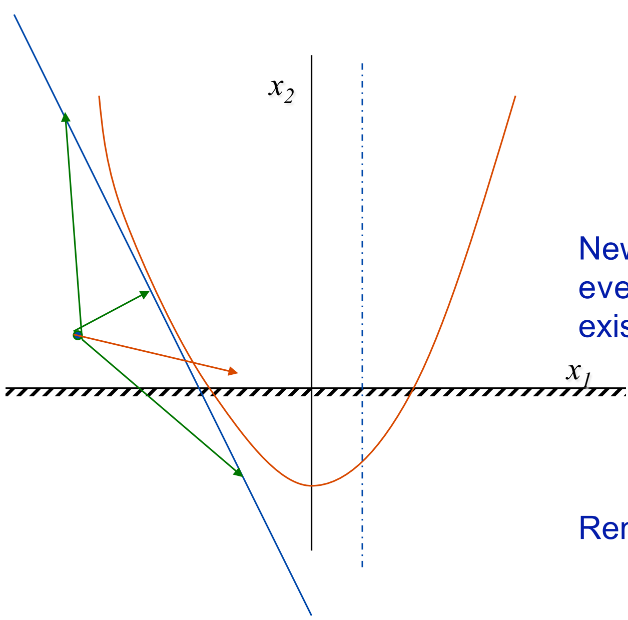
Line Search – choose *stepsize* α to give sufficient decrease of merit function using a 'step to the boundary' rule with τ ~0.99.

$$\text{for } \alpha \in (0, \bar{\alpha}],\ x_{k+1} = x_k + \alpha\, d_x$$
$$x_k + \bar{\alpha}\, d_x \geq (1-\tau) x_k > 0$$
$$v_{k+1} = v_k + \bar{\alpha}\, d_v \geq (1-\tau) v_k > 0$$
$$\lambda_{k+1} = \lambda_k + \alpha\,(\lambda_+ - \lambda_k)$$

- How do we balance $\phi(x)$ and $c(x)$ with η?
- Is this approach globally convergent? Will it still be fast?

Global Convergence Failure

(Wächter and B., 2000)



$$Min\ f(x)$$
$$s.t.\ x_1 - x_3 - \frac{1}{2} = 0$$
$$(x_1)^2 - x_2 - 1 = 0$$
$$x_2, x_3 \geq 0$$

Newton-type line search 'stalls' even though descent directions exist

$$A(x^k)^T d_x + c(x^k) = 0$$
$$x^k + \alpha\, d_x > 0$$

Remedies:

- •Composite Step Trust Region (Byrd et al.)
- •Filter Line Search Methods

Line Search Filter Method

Store (ϕ_k, θ_k) at allowed iterates

Allow progress if trial point is acceptable to filter with θ margin

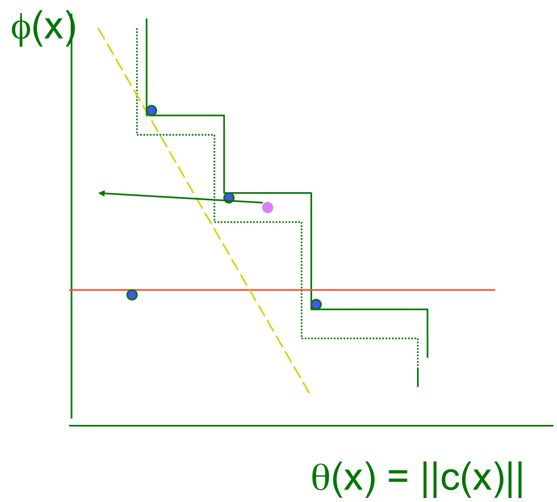
If switching condition

$$\alpha[-\nabla \phi_k^T d]^a \geq \delta[\theta_k]^b, a > 2b > 2$$

is satisfied, only an Armijo line search is required on ϕ_k

If insufficient progress on stepsize, evoke restoration phase to reduce θ.

Global convergence and superlinear local convergence proved (with second order correction)



Implementation Details

Modify KKT (full space) matrix if singular

$$\begin{bmatrix} W_k + \Sigma_k + \delta_1 & A_k \\ A_k^T & -\delta_2 I \end{bmatrix}$$

- δ_1 - Correct inertia to guarantee descent direction
- δ_2 - Deal with rank deficient A_k

KKT matrix factored by MA27

Feasibility restoration phase

$$Min \| c(x) \|_1 + \| x - x_k \|_Q^2$$

$$x_l \leq x_k \leq x_u$$

Apply Exact Penalty Formulation

Exploit same structure/algorithm to reduce infeasibility

IPOPT Algorithm – Features

Line Search Strategies for Globalization

- ℓ_2 exact penalty merit function
- augmented Lagrangian merit function
- **Filter method (adapted and extended from Fletcher and Leyffer)**

Hessian Calculation

- BFGS (full/LM and reduced space)
- SR1 (full/LM and reduced space)
- **Exact full Hessian (direct)**
- Exact reduced Hessian (direct)
- Preconditioned CG

Algorithmic Properties

Globally, superlinearly convergent (Wächter and B., 2005)

Easily tailored to different problem structures

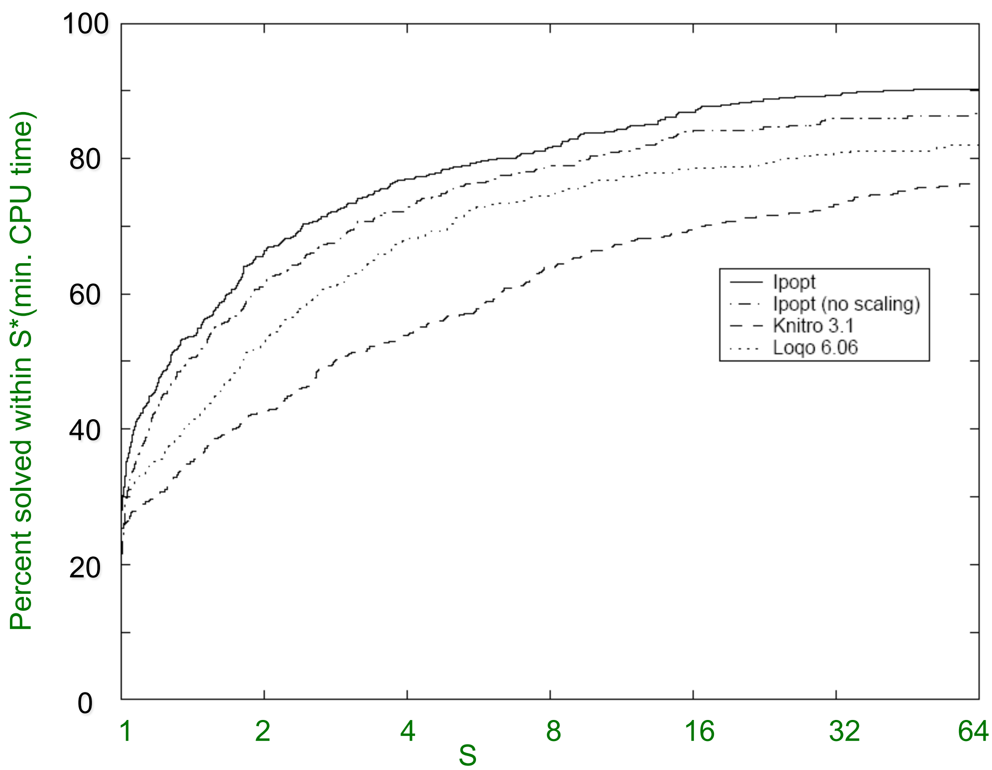
Freely Available

CPL License and COIN-OR distribution: **http://www.coin-or.org**

IPOPT 3.1 recently rewritten in C++

Solved on thousands of test problems and applications

IPOPT Comparison on 954 Test Problems



Recommendations for Constrained Optimization

1. Best current algorithms
   - GRG 2/CONOPT
   - MINOS
   - SQP
   - IPOPT
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

Fewer Function Evaluations ⟷ Tailored Linear Algebra

Small, Nonlinear Problems - SQP 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!
Exploit Both Features – IPOPT takes advantages of few function evaluations and large-scale linear algebra, but requires exact second derivatives

Available Software for Constrained Optimization

SQP Routines
HSL, NaG and IMSL (NLPQL) Routines
NPSOL – Stanford Systems Optimization Lab
SNOPT – Stanford Systems Optimization Lab (rSQP discussed later)
IPOPT – http://www.coin-or.org

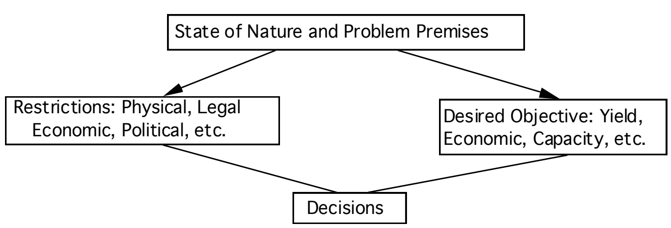
GAMS Programs
CONOPT - Generalized Reduced Gradient method with restoration
MINOS - Generalized Reduced Gradient method without restoration
A student version of GAMS is now available from the CACHE office. The cost for this package including Process Design Case Students, GAMS: A User's Guide, and GAMS - The Solver Manuals, and a CD-ROM is $65 per CACHE supporting departments, and $100 per non-CACHE supporting departments and individuals. To order please complete standard order form and fax or mail to CACHE Corporation. More information can be found on http://www.che.utexas.edu/cache/gams.html

MS Excel
Solver uses Generalized Reduced Gradient method with restoration

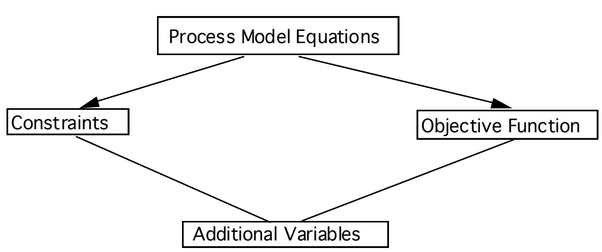
Rules for Formulating Nonlinear Programs

1) Avoid overflows and undefined terms, (do not divide, take logs, etc.)
   e.g. $x + y - \ln z = 0 \Rightarrow x + y - u = 0$
   $\exp u - z = 0$
2) If constraints must <u>always</u> be enforced, make sure they are linear or bounds.
   e.g. $v(xy - z^2)^{1/2} = 3 \Rightarrow vu = 3$
   $u^2 - (xy - z^2) = 0,\ u \geq 0$
3) Exploit linear constraints as much as possible, e.g. mass balance
   $$x_i L + y_i V = F z_i \Rightarrow l_i + v_i = f_i$$
   $$L - \sum l_i = 0$$
4) Use bounds and constraints to enforce characteristic solutions.
   e.g. $a \leq x \leq b,\ \ g(x) \leq 0$
   to isolate correct root of $h(x) = 0$.
5) Exploit <u>global</u> properties when possibility exists. Convex (linear equations?) Linear Program? Quadratic Program? Geometric Program?
6) Exploit problem structure when possible.
   e.g. $Min \quad [Tx - 3Ty]$
   $s.t. \quad xT + y - T^2 y = 5$
   $4x - 5Ty + Tx = 7$
   $0 \leq T \leq 1$
   (If T is fixed $\Rightarrow$ solve LP) $\Rightarrow$ put T in outer optimization loop.

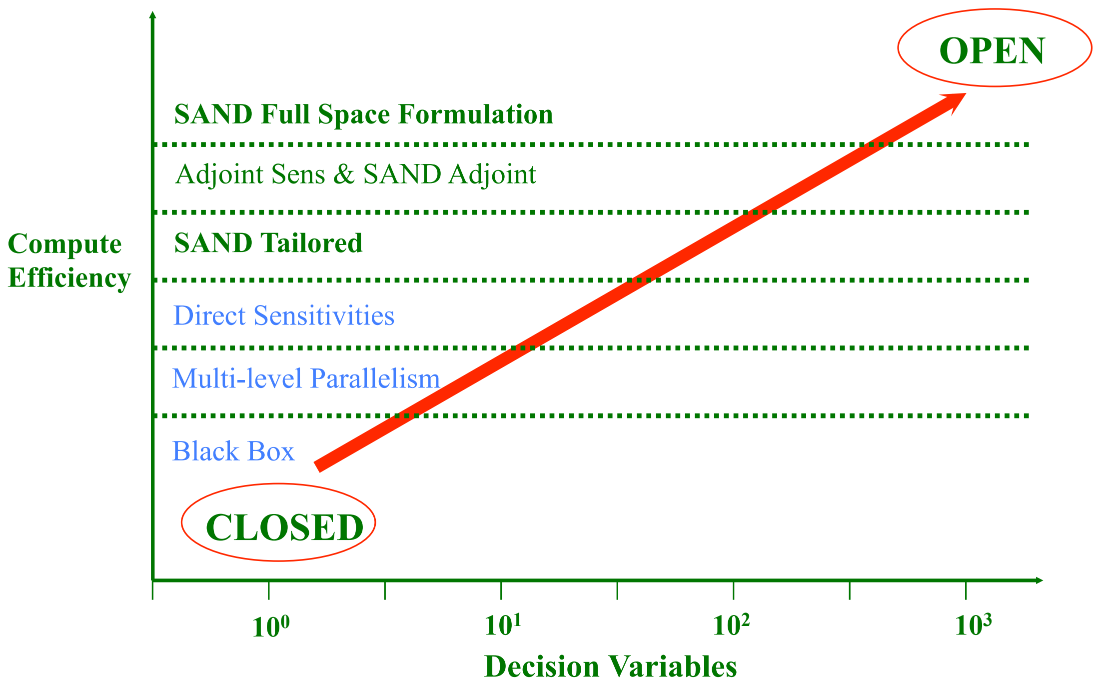
Process Optimization
Problem Definition and Formulation



Mathematical Modeling and Algorithmic Solution



Hierarchy of Nonlinear Programming Formulations and Model Intrusion



Large Scale NLP Algorithms

Motivation: Improvement of Successive Quadratic Programming as Cornerstone Algorithm

➔ ***process optimization for design, control and operations***

Evolution of NLP Solvers:

SQP → rSQP → IPOPT

rSQP ↓ rSQP++

IPOPT ↓ IPOPT 3.x

2000 - : Simultaneous dynamic optimization over 1 000 000 variables and constraints

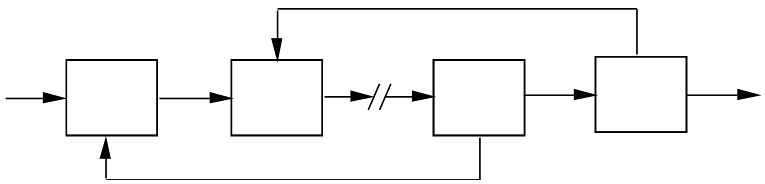
Current: Tailor structure, architecture and problems

Flowsheet Optimization Problems - Introduction

Modular Simulation Mode

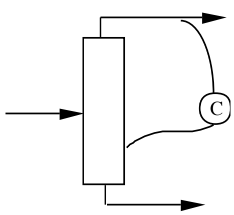
Physical Relation to Process

- Intuitive to Process Engineer
- Unit equations solved internally
- tailor-made procedures.



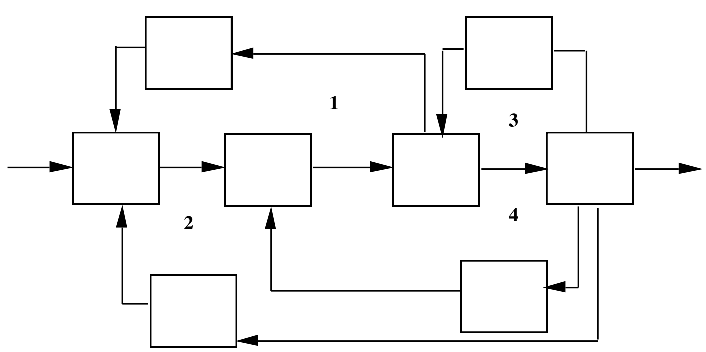
•Convergence Procedures - for simple flowsheets, often identified from flowsheet structure
•Convergence "mimics" startup.
•Debugging flowsheets on "physical" grounds

Flowsheet Optimization Problems - Features



Design Specifications

Specify # trays reflux ratio, but would like to specify overhead comp. ==> Control loop -Solve Iteratively



Nested Recycles Hard to Handle

Best Convergence Procedure?

•Frequent block evaluation can be expensive
•Slow algorithms applied to flowsheet loops.
•NLP methods are good at breaking loops

Chronology in Process Optimization

	Sim. Time Equiv.
1. Early Work - Black Box Approaches	
Friedman and Pinder (1972)	75-150
Gaddy and co-workers (1977)	300
2. Transition - more accurate gradients	
Parker and Hughes (1981)	64
Biegler and Hughes (1981)	13
3. Infeasible Path Strategy for Modular Simulators	
Biegler and Hughes (1982)	<10
Chen and Stadtherr (1985)	
Kaijaluoto et al. (1985)	
and many more	
4. Equation Based Process Optimization	
Westerberg et al. (1983)	<5
Shewchuk (1985)	2
DMO, NOVA, RTOPT, etc. (1990s)	1-2

Process optimization should be as cheap and easy as process simulation

Process Simulators with Optimization Capabilities (using SQP)

Aspen Custom Modeler (ACM)

Aspen/Plus

gProms

Hysim/Hysys

Massbal

Optisim

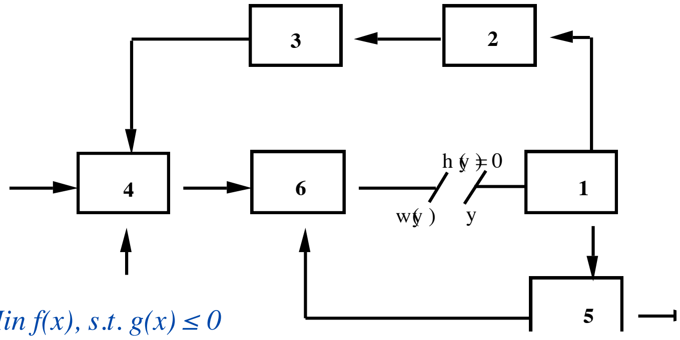
Pro/II

ProSim

ROMeo

VTPLAN

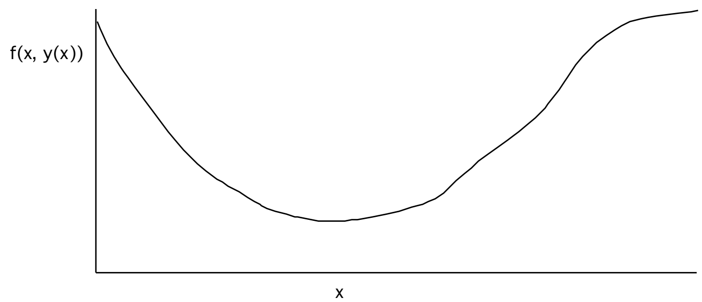
Simulation and Optimization of Flowsheets



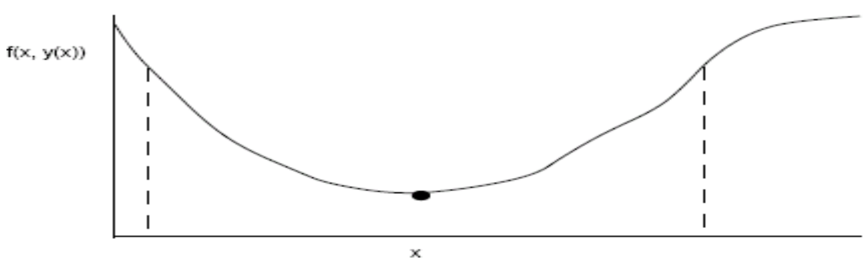
Min f(x), s.t. g(x) ≤ 0

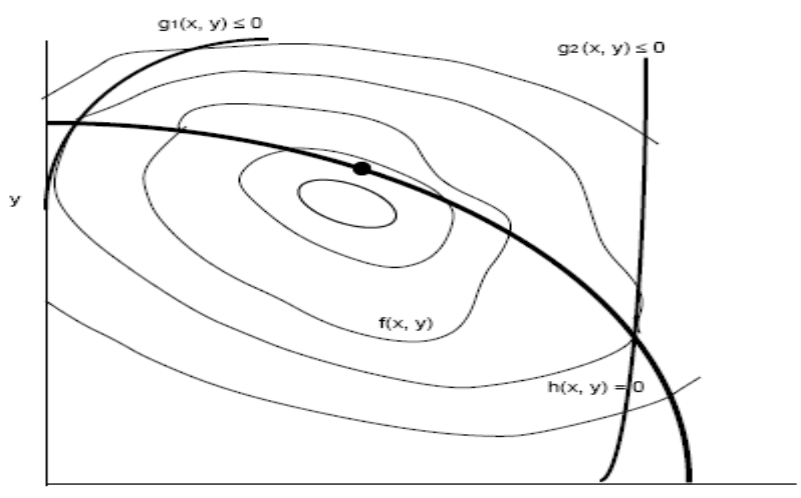
For single degree of freedom:

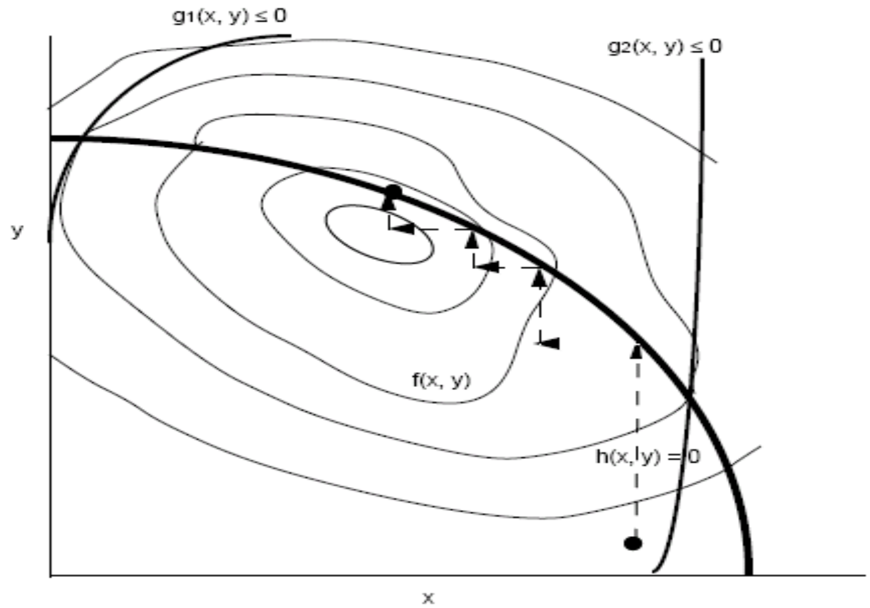
- work in space defined by curve below.
- requires repeated (expensive) recycle convergence



Expanded Region with Feasible Path

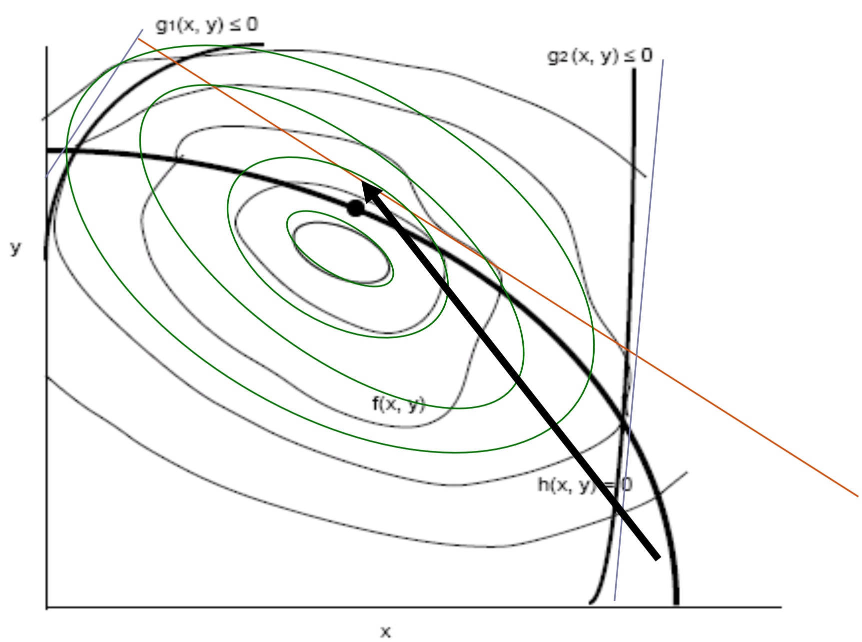






"Black Box" Optimization Approach

- Vertical steps are expensive (flowsheet convergence)
- Generally no connection between x and y.
- Can have "noisy" derivatives for gradient optimization.



SQP - Infeasible Path Approach

- solve and optimize simultaneously in x and y
- extended Newton method

Optimization Capability for Modular Simulators
(FLOWTRAN, Aspen/Plus, Pro/II, HySys)

Architecture

- Replace convergence with optimization block
- Problem definition needed (in-line FORTRAN)
- Executive, preprocessor, modules intact.

Examples

1. Single Unit and Acyclic Optimization
   - Distillation columns & sequences

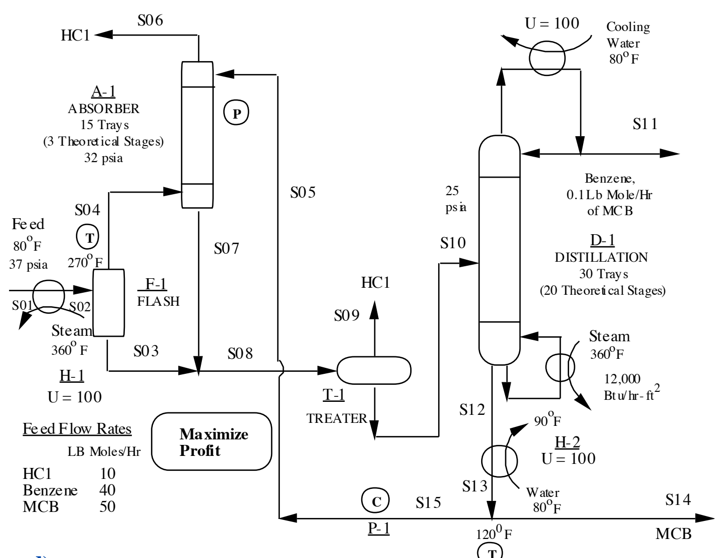
2. "Conventional" Process Optimization
   - Monochlorobenzene process
   - NH3 synthesis

3. Complicated Recycles & Control Loops
   - Cavett problem
   - Variations of above

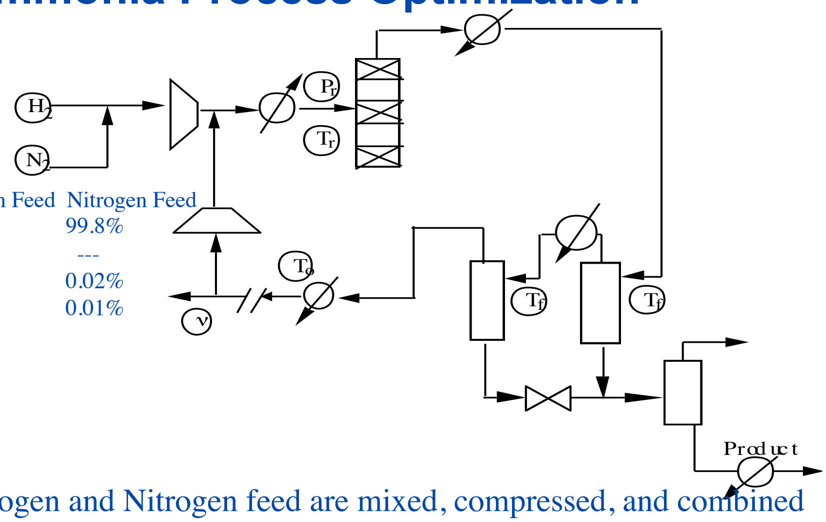
Optimization of Monochlorobenzene Process

PHYSICAL PROPERTY OPTIONS
Cavett Vapor Pressure
Redlich-Kwong Vapor Fugacity
Corrected Liquid Fugacity
Ideal Solution Activity Coefficient
OPT (SCOPT) OPTIMIZER
Optimal Solution Found After 4 Iterations
Kuhn-Tucker Error 0.29616E-05
Allowable Kuhn-Tucker Error 0.19826E-04
Objective Function -0.98259

Optimization Variables
32.006 0.38578 200.00 120.00
Tear Variables
0.10601E-19 13.064 79.229 120.00 50.000
Tear Variable Errors (Calculated Minus Assumed)
-0.10601E-19 0.72209E-06
-0.36563E-04 0.00000E+00 0.00000E+00
-Results of infeasible path optimization
-Simultaneous optimization and convergence of tear streams.



Ammonia Process Optimization



	Hydrogen Feed	Nitrogen Feed
N2	5.2%	99.8%
H2	94.0%	---
CH4	0.79 %	0.02%
Ar	---	0.01%

Hydrogen and Nitrogen feed are mixed, compressed, and combined with a recycle stream and heated to reactor temperature. Reaction occurs in a multibed reactor (modeled here as an equilibrium reactor) to partially convert the stream to ammonia. The reactor effluent is cooled and product is separated using two flash tanks with intercooling. Liquid from the second stage is flashed at low pressure to yield high purity NH_3 product. Vapor from the two stage flash forms the recycle and is recompressed.

Ammonia Process Optimization

Optimization Problem

Max {Total Profit @ 15% over five years}

s.t.
- 10^5 tons NH3/yr.
- Pressure Balance
- No Liquid in Compressors
- $1.8 \le$ H2/N2 ≤ 3.5
- Treact $\le 1000^o$ F
- NH3 purged $\le$ 4.5 lb mol/hr
- NH3 Product Purity $\ge$ 99.9 %
- Tear Equations

Performance Characterstics

- 5 SQP iterations.
- 2.2 base point simulations.
- objective function improves by $\$20.66 \times 10^6$ to $\$24.93 \times 10^6$.
- difficult to converge flowsheet at starting point

Item	Optimum	Starting point
Objective Function(10^6)	24.9286	20.659
1. Inlet temp. reactor (°F)	400	400
2. Inlet temp. 1st flash (°F)	65	65
3. Inlet temp. 2nd flash (°F)	35	35
4. Inlet temp. rec. comp. (°F)	80.52	107
5. Purge fraction (%)	0.0085	0.01
6. Reactor Press. (psia)	2163.5	2000
7. Feed 1 (lb mol/hr)	2629.7	2632.0
8. Feed 2 (lb mol/hr)	691.78	691.4

How accurate should gradients be for optimization?

Recognizing True Solution

- KKT conditions and Reduced Gradients determine true solution
- Derivative Errors will lead to wrong solutions!

Performance of Algorithms

Constrained NLP algorithms are gradient based
(SQP, Conopt, GRG2, MINOS, etc.)
Global and Superlinear convergence theory assumes accurate gradients

Worst Case Example (Carter, 1991)
Newton's Method generates an *ascent direction* and fails *for any* ε !

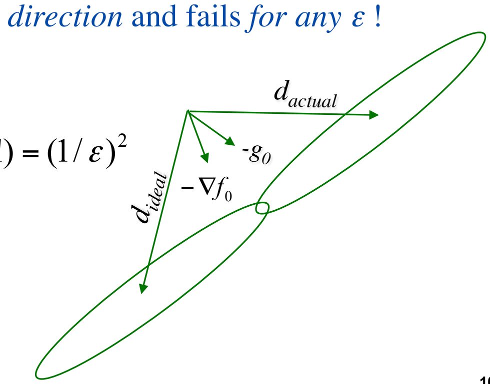
$$Min\ f(x) = x^T A x$$

$$A = \begin{bmatrix} \varepsilon + 1/\varepsilon & \varepsilon - 1/\varepsilon \\ \varepsilon - 1/\varepsilon & \varepsilon + 1/\varepsilon \end{bmatrix} \qquad \kappa(A) = (1/\varepsilon)^2$$

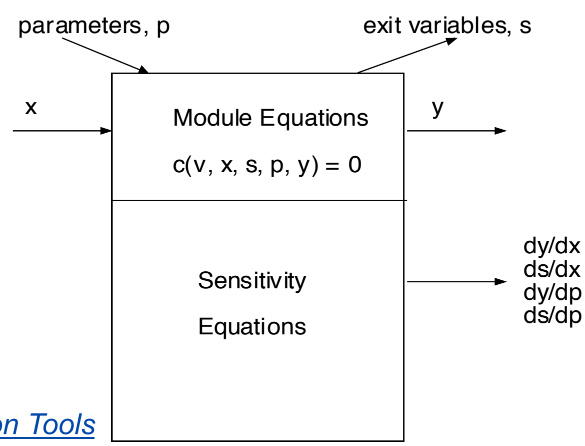
$$x_0 = [1 \quad 1]^T \quad \nabla f(x_0) = \varepsilon\, x_0$$

$$g(x_0) = \nabla f(x_0) + O(\varepsilon)$$

$$d = -A^{-1} g(x_0)$$



Implementation of Analytic Derivatives

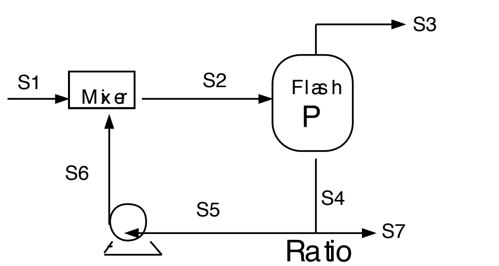


Automatic Differentiation Tools

JAKE-F, limited to a subset of FORTRAN (Hillstrom, 1982)
DAPRE, which has been developed for use with the NAG library (Pryce, Davis, 1987)
ADOL-C, implemented using operator overloading features of C++ (Griewank, 1990)
ADIFOR, (Bischof et al, 1992) uses source transformation approach FORTRAN code .
TAPENADE, web-based source transformation for FORTRAN code

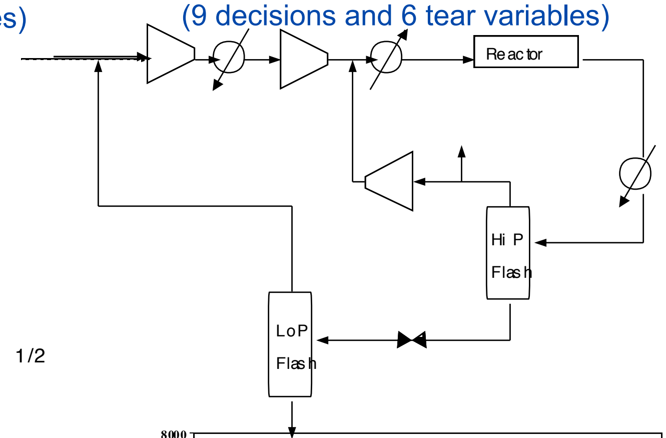
Relative effort needed to calculate gradients is not n+1 but about 3 to 5 (Wolfe, Griewank)

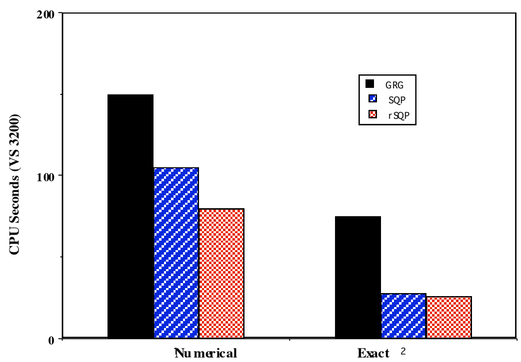
Flash Recycle Optimization
(2 decisions + 7 tear variables)

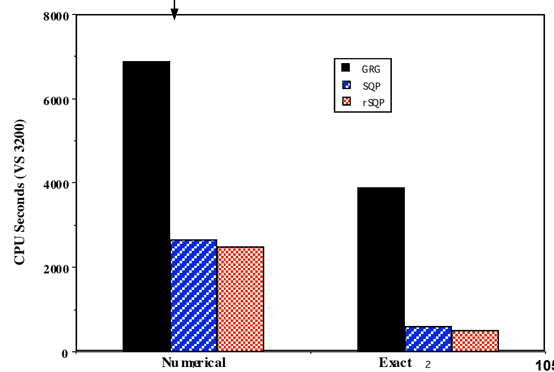


$$\text{Max } S3(A)^2 * S3(B) - S3(A)^2 - S3(C) + S3(D)^3 - (S3(E))^{1/2}$$

Ammonia Process Optimization
(9 decisions and 6 tear variables)







Large-Scale SQP

$$\begin{array}{ll} \min & f(z) \\ s.t. & c(z)=0 \\ & z_L \le z \le z_U \end{array} \qquad \begin{array}{ll} \min & \nabla f(z^k)^T d + 1/2\, d^T W^k d \\ s.t. & c(z^k) + (A^k)^T d = 0 \\ & z_L \le z^k + d \le z_U \end{array}$$

<u>Characteristics</u>
- Many equations and variables (≥ 100 000)
- Many bounds and inequalities (≥ 100 000)

<u>Few degrees of freedom (10 - 100)</u>
Steady state flowsheet optimization
Real-time optimization
Parameter estimation

<u>Many degrees of freedom (≥ 1000)</u>
Dynamic optimization (optimal control, MPC)
State estimation and data reconciliation

Few degrees of freedom => reduced space SQP (rSQP)

- Take advantage of sparsity of $A = \nabla c(x)$
- project W into space of active (or equality constraints)
- curvature (second derivative) information only needed in space of degrees of freedom
- second derivatives can be applied or approximated with positive curvature (e.g., BFGS)
- use dual space QP solvers

+ easy to implement with existing sparse solvers, QP methods and line search techniques
+ exploits 'natural assignment' of dependent and decision variables (some decomposition steps are 'free')
+ does not require second derivatives

- reduced space matrices are dense
- may be dependent on variable partitioning
- can be very expensive for many degrees of freedom
- can be expensive if many QP bounds

Reduced space SQP (rSQP)
Range and Null Space Decomposition

Assume no active bounds, QP problem with n variables and m constraints becomes:

$$\begin{bmatrix} W^k & A^k \\ A^{kT} & 0 \end{bmatrix} \begin{bmatrix} d \\ \lambda_+ \end{bmatrix} = -\begin{bmatrix} \nabla f(x^k) \\ c(x^k) \end{bmatrix}$$

- Define reduced space basis, $Z^k \in \Re^{n \times (n-m)}$ with $(A^k)^T Z^k = 0$
- Define basis for remaining space $Y^k \in \Re^{n \times m}$, $[Y^k\ Z^k] \in \Re^{n \times n}$ is nonsingular.
- Let $d = Y^k d_Y + Z^k d_Z$ to rewrite:

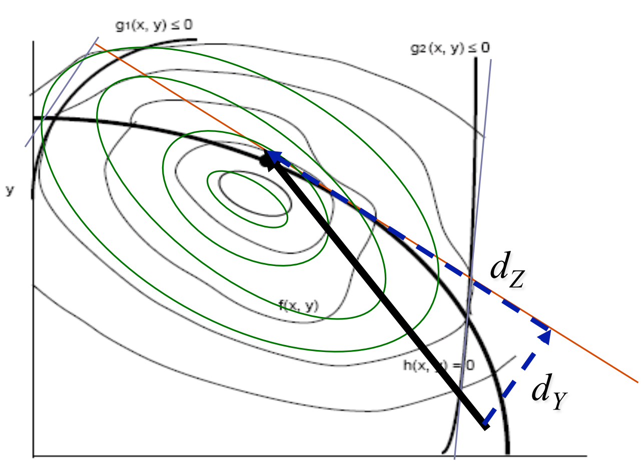
$$\begin{bmatrix} \begin{bmatrix} Y^k & Z^k \end{bmatrix}^T & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} W^k & A^k \\ A^{kT} & 0 \end{bmatrix} \begin{bmatrix} \begin{bmatrix} Y^k & Z^k \end{bmatrix} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} d_Y \\ d_Z \\ \lambda_+ \end{bmatrix} = -\begin{bmatrix} \begin{bmatrix} Y^k & Z^k \end{bmatrix}^T & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \nabla f(x^k) \\ c(x^k) \end{bmatrix}$$

Reduced space SQP (rSQP)
Range and Null Space Decomposition

$$\begin{bmatrix} \cancelto{0}{Y^{k^T} W^k Y^k} & \cancelto{0}{Y^{k^T} W^k Z^k} & Y^{k^T} A^k \\ Z^{k^T} W^k Y^k & Z^{k^T} W^k Z^k & 0 \\ A^{k^T} Y^k & 0 & 0 \end{bmatrix} \begin{bmatrix} d_Y \\ d_Z \\ \lambda_+ \end{bmatrix} = - \begin{bmatrix} Y^{k^T} \nabla f(x^k) \\ Z^{k^T} \nabla f(x^k) \\ c(x^k) \end{bmatrix}$$

- $\boldsymbol{(A^T Y)\, d_Y = -c(x^k)}$ is square, d_Y determined from bottom row.
- Cancel $Y^T W Y$ and $Y^T W Z$; (unimportant as d_Z, $d_Y \to 0$)
- $\boldsymbol{(Y^T A)\, \lambda = -Y^T \nabla f(x^k)}$, λ can be determined by first order estimate
- Calculate or approximate $w = Z^T W Y\, d_Y$, solve $\boldsymbol{Z^T W Z\, d_Z = -Z^T \nabla f(x^k) - w}$
- Compute total step: $\boldsymbol{d = Y\, d_Y + Z\, d_Z}$

Reduced space SQP (rSQP) Interpretation



Range and Null Space Decomposition

- SQP step (d) operates in a higher dimension
- Satisfy constraints using range space to get d_Y
- Solve small QP in null space to get d_Z
- In general, same convergence properties as SQP.

Choice of Decomposition Bases

1. Apply QR factorization to A. Leads to dense but well-conditioned Y and Z.

$$A = Q\begin{bmatrix} R \\ 0 \end{bmatrix} = \begin{bmatrix} Y & Z \end{bmatrix}\begin{bmatrix} R \\ 0 \end{bmatrix}$$

2. Partition variables into decisions u and dependents v. Create orthogonal Y and Z with embedded identity matrices ($A^TZ = 0,\ Y^TZ=0$).

$$A^T = \begin{bmatrix} \nabla_u c^T & \nabla_v c^T \end{bmatrix} = \begin{bmatrix} N & C \end{bmatrix}$$

$$Z = \begin{bmatrix} I \\ -C^{-1}N \end{bmatrix} \quad Y = \begin{bmatrix} N^T C^{-T} \\ I \end{bmatrix}$$

3. Coordinate Basis - same Z as above, $Y^T = [\ 0 \quad I\ ]$

- Bases use gradient information already calculated.
- Adapt decomposition to QP step
- Theoretically same rate of convergence as original SQP.
- Coordinate basis can be sensitive to choice of u and v. Orthogonal is not.
- Need consistent initial point and nonsingular C; automatic generation

rSQP Algorithm

1. Choose starting point x^0.
2. At iteration k, evaluate functions $f(x^k)$, $c(x^k)$ and their gradients.
3. Calculate bases Y and Z.
4. Solve for step d_Y in Range space from
   $(A^TY)\ d_Y = -c(x^k)$
5. Update projected Hessian $B^k \sim Z^TWZ$ (e.g. with BFGS), w_k (e.g., zero)
6. Solve small QP for step d_Z in Null space.

$$\begin{aligned} &Min \quad (Z^T \nabla f(x^k) + w^k)^T d_Z + 1/2 d_Z{}^T B^k d_Z \\ &s.t. \quad x_L \le x^k + Yd_Y + Zd_Z \le x_U \end{aligned}$$

7. If error is less than tolerance stop. Else
8. Solve for multipliers using $(Y^TA)\ \lambda = -Y^T \nabla f(x^k)$
9. Calculate total step $d = Y\,d_Y + Z\,d_Z$.
10. Find step size α and calculate new point, $x_{k+1} = x_k + \alpha\, d$
13. Continue from step 2 with $k = k+1$.

rSQP Results: Computational Results for General Nonlinear Problems

Vasantharajan et al (1990)

Problem	Specifications			MINOS (5.2)		Reduced SQP	
	N	M	MEQ	TIME*	FUNC	TIME* RND/LP	FUNC
Ramsey	34	23	10	1.4	46	1.7 1.0/0.7	8
Chenery	44	39	20	2.6	81	4.6 2.1/2.5	18
Korcge	100	96	78	3.9	9	3.7 1.4/2.3	3
Camcge	280	243	243	23.6	14	24.4 10.3/14.1	3
Ganges	357	274	274	22.7	14	59.7 35.7/24.0	4

* CPU Seconds - VAX 6320

rSQP Results: Computational Results for Process Problems

Vasantharajan et al (1990)

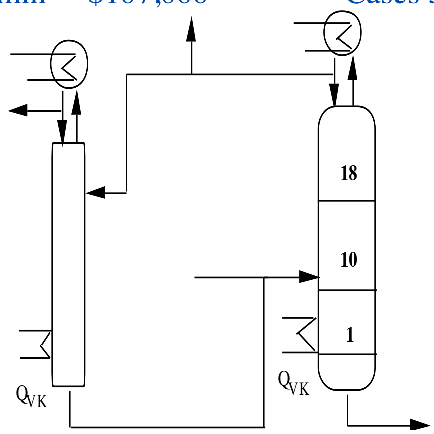
Prob.	Specifications			MINOS (5.2)		Reduced SQP	
	N	M	MEQ	TIME*	FUNC	TIME* (rSQP/LP)	FUN.
Absorber (a) (b)	50	42	42	 4.4 4.7	 144 157	 3.2 (2.1/1.1) 2.8 (1.6/1.2)	 23 13
Distill'n Ideal (a) (b)	228	227	227	 28.5 33.5	 24 58	 38.6 (9.6/29.0) 69.8 (17.2/52.6)	 7 14
Distill'n Nonideal (1) (a) (b) (c)	569	567	567	 172.1 432.1 855.3	 34 362 745	 130.1 (47.6/82.5) 144.9 (132.6/12.3) 211.5 (147.3/64.2)	 14 47 49
Distill'n Nonideal (2) (a) (b) (c)	977	975	975	 (F) 520.0$^+$ (F)	 (F) 162 (F)	 230.6 (83.1/147.5) 322.1 (296.4/25.7) 466.7 (323/143.7)	 9 26 34

* CPU Seconds - VAX 6320 (F) Failed
+ MINOS (5.1)

Comparison of SQP and rSQP

Coupled Distillation Example - 5000 Equations
Decision Variables - boilup rate, reflux ratio

	Method	CPU Time	Annual Savings	Comments
1.	SQP*	2 hr	negligible	Base Case
2.	rSQP	15 min.	$ 42,000	Base Case
3.	rSQP	15 min.	$ 84,000	Higher Feed Tray Location
4.	rSQP	15 min.	$ 84,000	Column 2 Overhead to Storage
5.	rSQP	15 min	$107,000	Cases 3 and 4 together

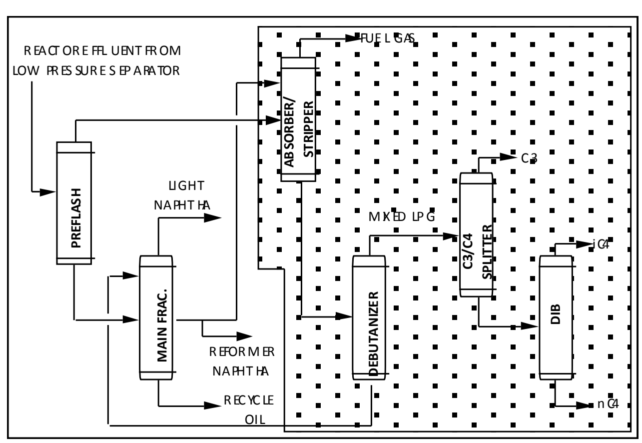


Real-time Optimization with rSQP

Sunoco Hydrocracker Fractionation Plant

(Bailey et al, 1993)

Existing process, optimization on-line at regular intervals: 17 hydrocarbon components, 8 heat exchangers, absorber/stripper (30 trays), debutanizer (20 trays), C3/C4 splitter (20 trays) and deisobutanizer (33 trays).



- square *parameter case* to fit the model to operating data.
- optimization to determine best operating conditions

Optimization Case Study Characteristics

Model consists of 2836 equality constraints and only ten independent variables. It is also reasonably sparse and contains 24123 nonzero Jacobian elements.

$$P = \sum_{i \in G} z_i C_i^{G} + \sum_{i \in E} z_i C_i^{E} + \sum_{m=1}^{NP} z_i C_i^{P_m} - U$$

<u>Cases Considered:</u>

1. Normal Base Case Operation
2. Simulate fouling by reducing the heat exchange coefficients for the debutanizer
3. Simulate fouling by reducing the heat exchange coefficients for splitter feed/bottoms exchangers
4. Increase price for propane
5. Increase base price for gasoline together with an increase in the octane credit

	Case 0 Base Parameter	Case 1 Base Optimization	Case 2 Fouling 1	Case 3 Fouling 2	Case 4 Changing Market 1	Case 5 Changing Market 2
Heat Exchange Coefficient (TJ/d∞C)						
Debutanizer Feed/Bottoms	$6.565\text{x}10^{-4}$	$6.565\text{x}10^{-4}$	$5.000\text{x}10^{-4}$	$2.000\text{x}10^{-4}$	$6.565\text{x}10^{-4}$	$6.565\text{x}10^{-4}$
Splitter Feed/Bottoms	$1.030\text{x}10^{-3}$	$1.030\text{x}10^{-3}$	$5.000\text{x}10^{-4}$	$2.000\text{x}10^{-4}$	$1.030\text{x}10^{-3}$	$1.030\text{x}10^{-3}$
Pricing						
Propane ($/m^3)	180	180	180	180	300	180
Gasoline Base Price ($/m^3	300	300	300	300	300	350
Octane Credit ($/(RON m^3))	2.5	2.5	2.5	2.5	2.5	10
Profit	230968.96	239277.37	239267.57	236706.82	258913.28	370053.98
Change from base case ($/d, %)	-	8308.41 (3.6%)	8298.61 (3.6%)	5737.86 (2.5%)	27944.32 (12.1%)	139085.02 (60.2%)
Infeasible Initialization						
MINOS						
Iterations (Major/Minor)	5 / 275	9 /788	-	-	-	-
CPU Time (s)	182	5768	-	-	-	-
rSQP						
Iterations	5	20	12	24	17	12
CPU Time (s)	23.3	80.1	54.0	93.9	69.8	54.2
Parameter Initialization						
MINOS						
Iterations (Major/Minor)	n/a	12 / 132	14 / 120	16 / 156	11 / 166	11 / 76
CPU Time (s)	n/a	462	408	1022	916	309
rSQP						
Iterations	n/a	13	8	18	11	10
CPU Time (s)	n/a	58.8	43.8	74.4	52.5	49.7
Time rSQP / Time MINOS (%)	12.8%	12.7%	10.7%	7.3%	5.7%	16.1%

Nonlinear Optimization Engines

Evolution of NLP Solvers:

➔ ***process optimization for design, control and operations***

SQP ⟶ rSQP ⟶ IPOPT

'00s: Simultaneous dynamic optimization
over 1 000 000 variables and constraints

Many degrees of freedom => full space IPOPT

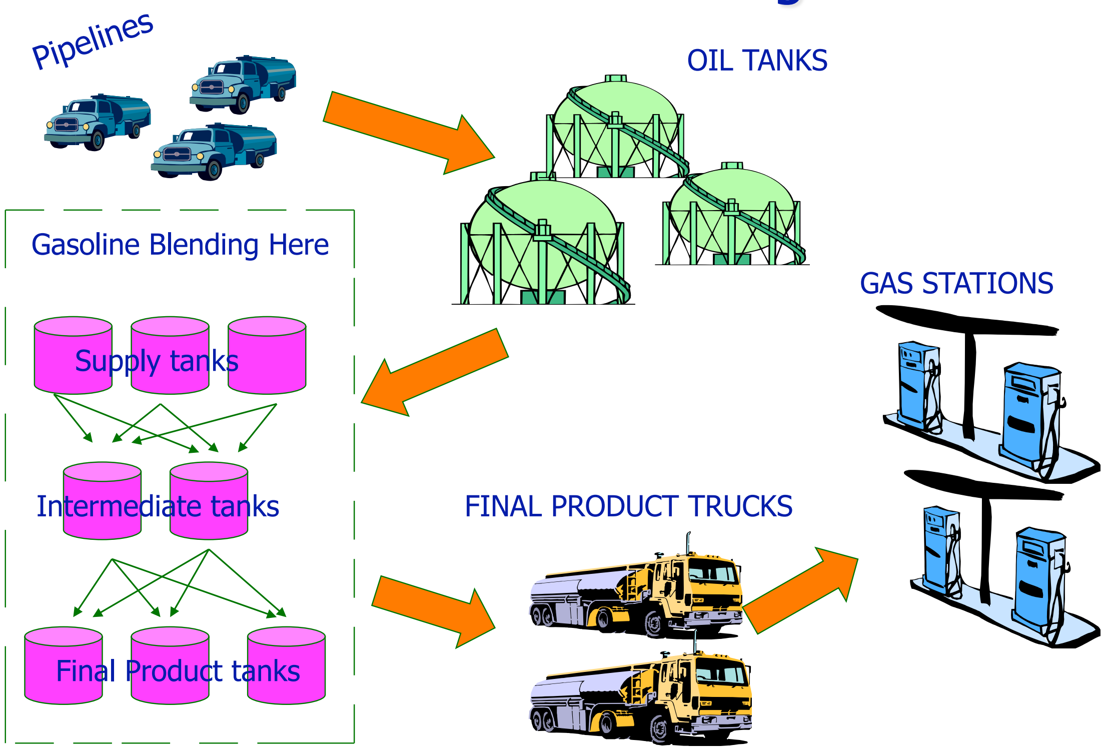
$$\begin{bmatrix} W^k + \Sigma & A^k \\ A^{k^T} & 0 \end{bmatrix} \begin{bmatrix} d \\ \lambda_+ \end{bmatrix} = -\begin{bmatrix} \nabla \varphi(x^k) \\ c(x^k) \end{bmatrix}$$

- work in full space of all variables
- second derivatives useful for objective and constraints
- use specialized large-scale Newton solver

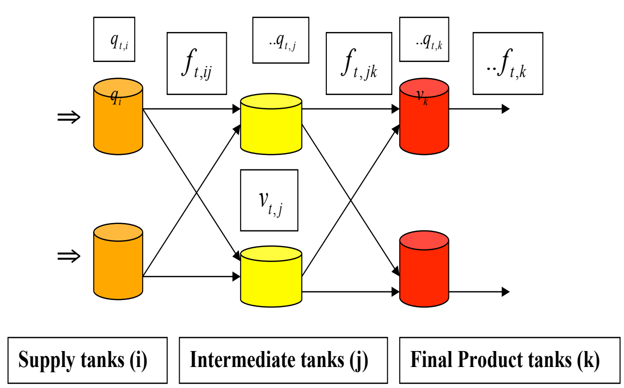
+ $W = \nabla_{xx} L(x,\lambda)$ and $A = \nabla c(x)$ sparse, often structured
+ fast if many degrees of freedom present
+ no variable partitioning required

- second derivatives strongly desired
- W is indefinite, requires complex stabilization
- requires specialized large-scale linear algebra

Gasoline Blending



Blending Problem & Model Formulation



$$\max \sum_t \left(\sum_k c_k f_{t,k} - \sum_i c_i f_{t,i}\right)$$

$$\text{s.t.} \quad \sum_k f_{t,jk} - \sum_i f_{t,ij} + v_{t+1,j} = v_{t,j}$$

$$f_{t,k} - \sum_j f_{t,jk} = 0$$

$$\sum_k q_{t,j} f_{t,jk} - \sum_i q_{t,i} f_{t,ij} + q_{t+1,j} v_{t+1,j} = q_{t,j} v_{t,j}$$

$$q_{t,k} f_{t,k} - \sum_j q_{t,j} f_{t,jk} = 0$$

$$q_{k_{\min}} \le q_{t,k} \le q_{k_{\max}}$$

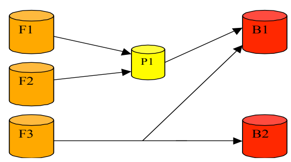
$$v_{j_{\min}} \le v_{t,j} \le v_{j_{\max}}$$

f & v ------ flowrates and tank volumes
q ------ tank qualities

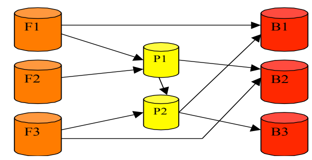
Model Formulation in AMPL

Small Multi-day Blending Models
Single Qualities

Haverly, C. 1978 (HM)



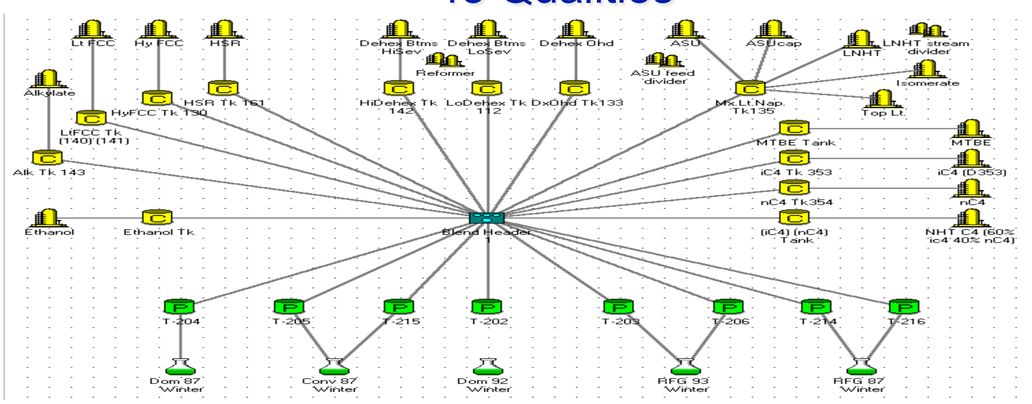
Audet & Hansen 1998 (AHM)



	no. of iterations	objective	CPU (s)	normalized CPU (s)
HM Day 1 ($N = 13$, $M = 8$, $S = 8$)				
LANCELOT	62	100	0.10	0.05
MINOS	15	400	0.04	0.13
SNOPT	36	400	0.02	0.01
KNITRO	38	100	0.14	0.06
LOQO	30	400	0.10	0.08
IPOPT, exact	31	400	0.01	0.01
IPOPT, L-BFGS	199	400	0.08	0.08
AHM Day 1 ($N = 21$, $M = 14$, $S = 14$)				
LANCELOT	112	49.2	0.32	0.14
MINOS	29	0.00	0.01	0.03
SNOPT	60	49.2	0.01	<0.01
KNITRO	44	31.6	0.15	0.07
LOQO	28	49.2	0.10	0.08
IPOPT, exact	28	49.2	0.01	0.01
IPOPT, L-BFGS	44	49.2	0.02	0.02

	no. of iterations	objective	CPU (s)	normalized CPU (s)
HM Day 25 ($N = 325$, $M = 200$, $S = 200$)				
LANCELOT	67	1.00×10^4	6.75	3.04
MINOS	801	6.40×10^3	1.21	3.83
SNOPT	739	1.00×10^4	0.59	0.27
KNITRO	>1000	*a*	*a*	*a*
LOQO	31	1.00×10^4	0.44	0.33
IPOPT, exact	47	1.00×10^4	0.24	0.24
IPOPT, L-BFGS	344	1.00×10^4	1.99	1.99
AHM Day 25 ($N = 525$, $M = 300$, $S = 350$)				
LANCELOT	149	8.13×10^2	26.8	12.1
MINOS	940	3.75×10^2	2.92	9.23
SNOPT	1473	1.23×10^3	1.47	0.66
KNITRO	316	1.13×10^3	17.5	7.88
LOQO	30	1.23×10^3	0.80	0.60
IPOPT, exact	44	1.23×10^3	0.25	0.25
IPOPT, L-BFGS	76	1.23×10^3	0.98	0.98

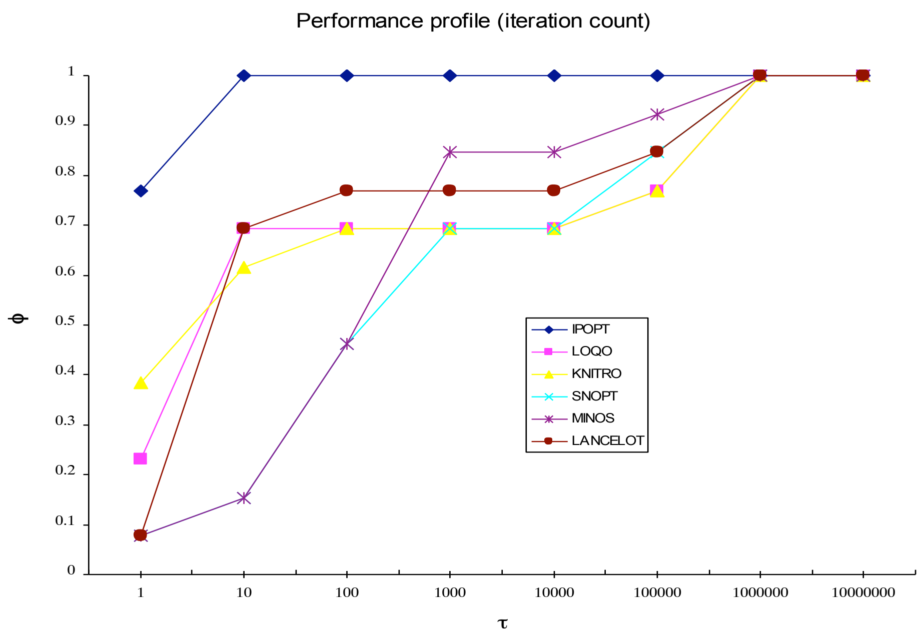
Honeywell Blending Model – Multiple Days
48 Qualities



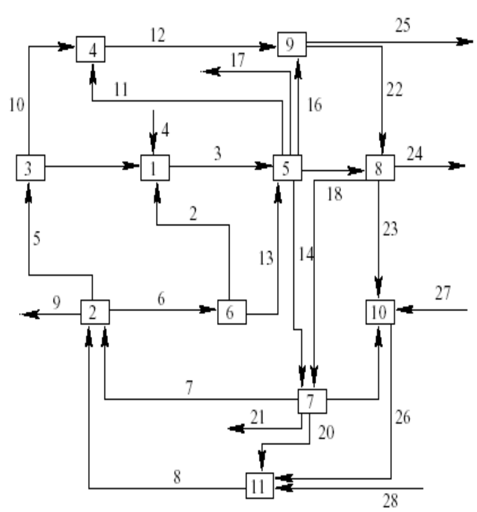
	no. of iterations	objective	CPU (s)	normalized CPU (s)
IHM Day 1 ($N = 2003$, $M = 1595$, $S = 1449$)				
LANCELOT	388	6.14×10^1	1.17×10^5	5.28×10^3
MINOS	2238	6.14×10^1	5.24×10^1	1.66×10^2
SNOPT	*a*	*a*	*a*	*a*
KNITRO	37	1.00×10^2	1.58×10^2	7.11×10^1
LOQO	*b*	*b*	*b*	*b*
IPOPT, exact	21	6.14×10^1	2.60	2.60
IPOPT, L-BFGS	52	6.14×10^1	8.89	8.89
IHM Day 5 ($N = 10\,134$, $M = 8073$, $S = 7339$)				
LANCELOT	*c*	*c*	*c*	*c*
MINOS	8075	1.39×10^5	3.08×10^2	9.74×10^2
SNOPT	*a*	*a*	*a*	*a*
KNITRO	*a*	*a*	*a*	*a*
LOQO	*b*	*b*	*b*	*b*
IPOPT, exact	39	1.39×10^5	1.06×10^3	1.06×10^3
IPOPT, L-BFGS	1000	1.39×10^5	2.91×10^5	2.91×10^5

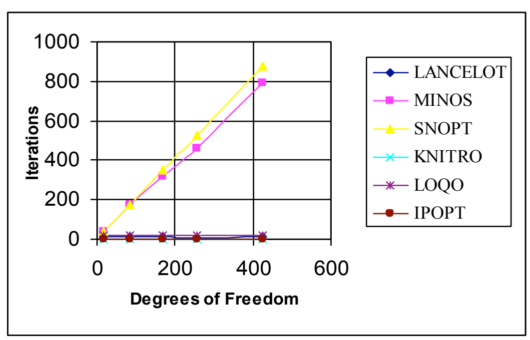
	no. of iterations	objective	CPU (s)	normalized CPU (s)
IHM Day 10 ($N = 20\,826$, $M = 16\,074$, $S = 15\,206$)				
LANCELOT	*c*	*c*	*c*	*c*
MINOS	*a*	*a*	*a*	*a*
SNOPT	*a*	*a*	*a*	*a*
KNITRO	*a*	*a*	*a*	*a*
LOQO	*b*	*b*	*b*	*b*
IPOPT, exact	65	2.64×10^4	1.12×10^4	1.12×10^4
IHM Day 15 ($N = 31\,743$, $M = 25\,560$, $S = 23\,073$)				
LANCELOT	*c*	*c*	*c*	*c*
MINOS	*a*	*a*	*a*	*a*
SNOPT	*a*	*a*	*a*	*a*
KNITRO	*a*	*a*	*a*	*a*
LOQO	*b*	*b*	*b*	*b*
IPOPT, exact	110	4.15×10^4	7.25×10^4	7.25×10^4

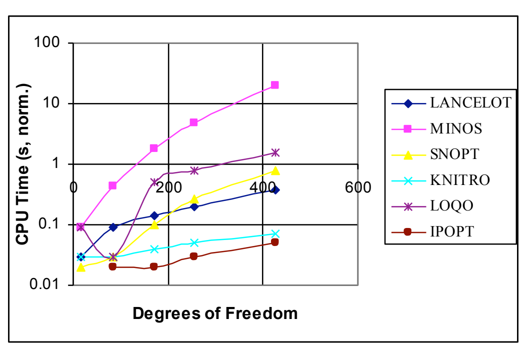
Summary of Results – Dolan-Moré plot



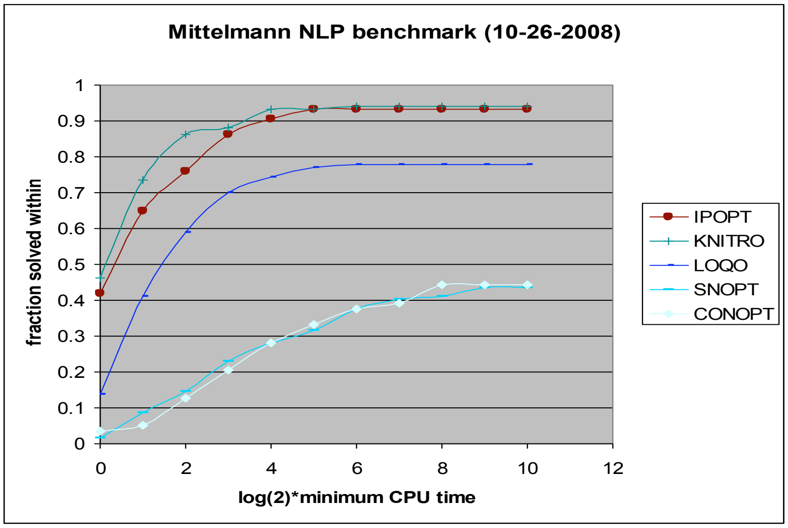
Comparison of NLP Solvers: Data Reconciliation







Comparison of NLP solvers
(latest Mittelmann study)



	Limits	Fail
IPOPT	7	2
KNITRO	7	0
LOQO	23	4
SNOPT	56	11
CONOPT	55	11

117 Large-scale Test Problems

500 - 250 000 variables, 0 – 250 000 constraints

Typical NLP algorithms and software

SQP - NPSOL, VF02AD, NLPQL, fmincon

reduced SQP - SNOPT, rSQP, MUSCOD, DMO, LSSOL…

Reduced Grad. rest. - GRG2, GINO, SOLVER, CONOPT

Reduced Grad no rest. - MINOS

Second derivatives and barrier - IPOPT, KNITRO, LOQO

Interesting hybrids -

•FSQP/cFSQP - SQP and constraint elimination

•LANCELOT (Augmented Lagrangian w/ Gradient Projection)

Sensitivity Analysis for Nonlinear Programming

At nominal conditions, p_0

$$\begin{aligned} & Min\ f(x, p_0) \\ s.t.\quad & c(x, p_0) = 0 \\ & a(p_0) \le x \le b(p_0) \end{aligned}$$

How is the optimum affected at other conditions, $p \neq p_0$?

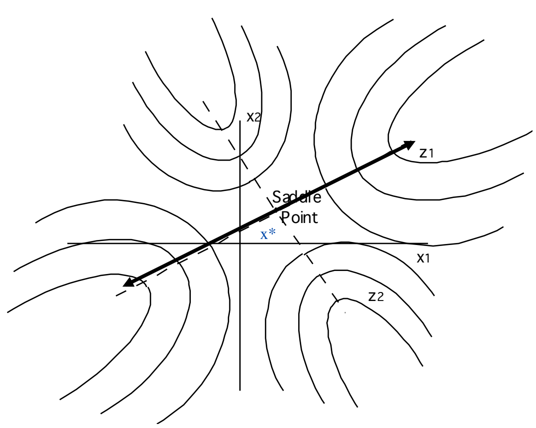
- Model parameters, prices, costs
- Variability in external conditions
- Model structure

- How sensitive is the optimum to parametric uncertainties?
- Can this be analyzed easily?

Second Order Optimality Conditions:
Reduced Hessian needs to be positive semi-definite

- Nonstrict local minimum: If nonnegative, find eigenvectors for zero eigenvalues, ➔ regions of nonunique solutions

- Saddle point: If any are eigenvalues are negative, move along directions of corresponding eigenvectors and restart optimization.



IPOPT Factorization Byproducts: Tools for Postoptimality and Uniqueness

Modify KKT (full space) matrix if nonsingular

$$\begin{bmatrix} W_k + \Sigma_k + \delta_1 I & A_k \\ A_k^T & -\delta_2 I \end{bmatrix}$$

- δ_1 - Correct inertia to guarantee descent direction
- δ_2 - Deal with rank deficient A_k

KKT matrix factored by indefinite symmetric factorization

•Solution with $\delta_1, \delta_2 = 0$ ➔ sufficient second order conditions

•Eigenvalues of reduced Hessian all positive – unique minimizer and multipliers

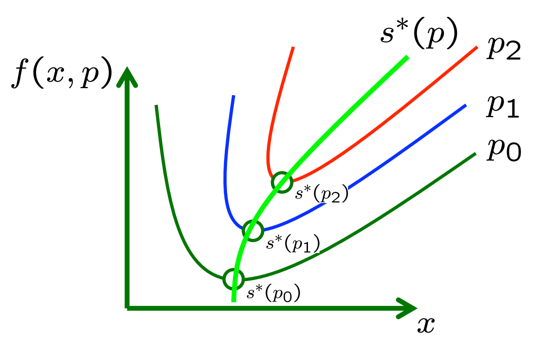
•Else:

- Reduced Hessian available through sensitivity calculations
- Find eigenvalues to determine nature of stationary point

NLP Sensitivity

Parametric Programming

$$\left.\begin{array}{ll} \min & f(x,p) \\ \text{s.t.} & c(x,p) = 0 \\ & x \geq 0 \end{array}\right\} \mathbf{P}(p)$$



Solution Triplet

$s^*(p)^T = [x^{*T}\ \lambda^{*T}\ \nu^{*T}]$

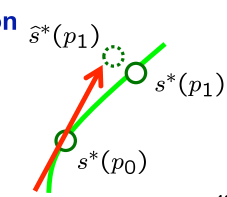
Optimality Conditions $\mathbf{P}(p)$

$$\begin{aligned} \nabla_x f(x,p) + \nabla_x c(x,p)\,\lambda - \nu &= 0 \\ c(x,p) &= 0 \\ XVe &= 0 \end{aligned}$$

NLP Sensitivity → Rely upon Existence and Differentiability of $s^*(p)$

→ Main Idea: Obtain $\left.\frac{\partial s}{\partial p}\right|_{p_0}$ **and find** $\hat{s}^*(p_1)$ **by Taylor Series Expansion**

$$\hat{s}^*(p_1) \approx s^*(p_0) + \left.\frac{\partial s}{\partial p}^T\right|_{p_0} (p_1 - p_0)$$



NLP Sensitivity Properties (Fiacco, 1983)

Assume sufficient differentiability, LICQ, SSOC, SC:

Intermediate IP solution $(s(\mu)-s^*) = O(\mu)$

Finite neighborhood around p_0 and $\mu=0$ with same active set

$$\left.\frac{\partial s}{\partial p}\right|_{p_0}$$ exists and is unique

$$s(p) - [s(p_0) + \left.\frac{\partial s}{\partial p}\right|_{p_0}^T (p - p_0)] = O((p-p_0)^2)$$

$$s(p) - [s(p_0, \mu) + \left.\frac{\partial s}{\partial p}\right|_{p_0,\mu}^T (p - p_0)] = O((p-p_0)^2) + O(\mu)$$

NLP Sensitivity

Obtaining $\left.\frac{\partial s}{\partial p}\right|_{p_0}$

Optimality Conditions of $\mathbf{P}(p)$

$$\left.\begin{aligned} \nabla_x \mathcal{L} = \nabla_x f(x,p) + \nabla_x c(x,p)\,\lambda - \nu &= 0 \\ c(x,p) &= 0 \\ XVe &= 0 \end{aligned}\right\} \quad \mathbf{Q}(s,p) = 0$$

Apply Implicit Function Theorem to $\mathbf{Q}(s,p) = 0$ **around** $(p_0, s^*(p_0))$

$$\underbrace{\frac{\partial \mathbf{Q}(s^*(p_0),p_0)}{\partial s} \left.\frac{\partial s}{\partial p}\right|_{p_0} + \frac{\partial \mathbf{Q}(s^*(p_0),p_0)}{\partial p} = 0}$$

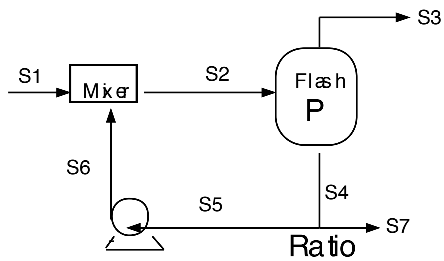
$$\begin{bmatrix} W(s^*(p_0)) & A(x^*(p_0)) & -I \\ A(x^*(p_0))^T & 0 & 0 \\ V^*(p_0) & 0 & X^*(p_0) \end{bmatrix} \begin{bmatrix} \frac{\partial x}{\partial p} \\ \frac{\partial \lambda}{\partial p} \\ \frac{\partial \nu}{\partial p} \end{bmatrix} + \begin{bmatrix} \nabla_{x,p}\mathcal{L}(s^*(p_0)) \\ \nabla_p\, c(x^*(p_0)) \\ 0 \end{bmatrix} = 0$$

KKT Matrix IPOPT

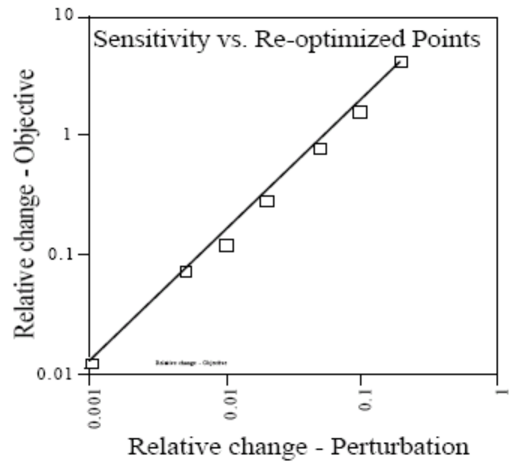
$$\begin{bmatrix} W(x_k,\lambda_k) & A(x_k) & -I \\ A(x_k)^T & 0 & 0 \\ V_k & 0 & X_k \end{bmatrix}$$

- → **Already Factored at Solution**
- → **Sensitivity Calculation from Single Backsolve**
- → **Approximate Solution Retains Active Set**

Sensitivity for Flash Recycle Optimization
(2 decisions, 7 tear variables)

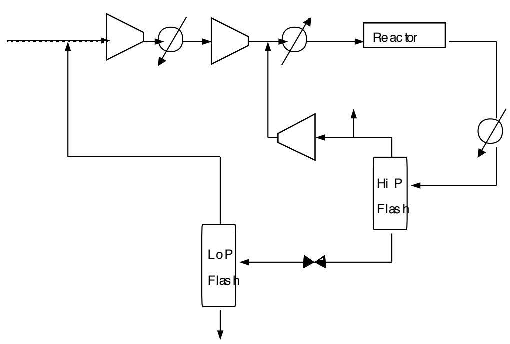


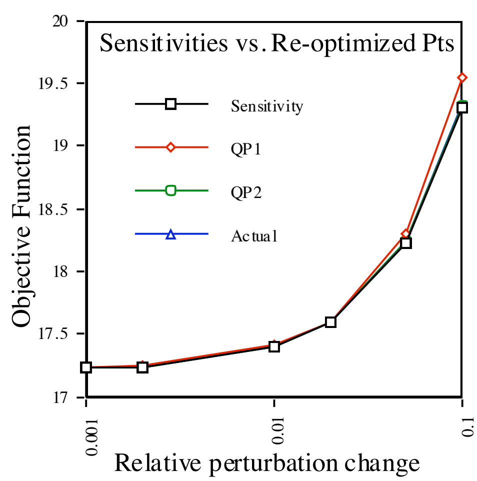
Max $S3(A)^2 * S3(B) - S3(A)^2 - S3(C)^3 + S3(D) - (S3(E))^{1/2}$



- Second order sufficiency test:
- Dimension of reduced Hessian = 1
- Positive eigenvalue
- Sensitivity to simultaneous change in feed rate and upper bound on purge ratio

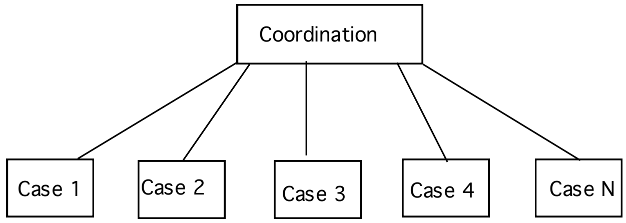
Ammonia Process Optimization
(9 decisions, 8 tear variables)





- Second order sufficiency test:
- Dimension of reduced Hessian = 4
- Eigenvalues = [2.8E-4, 8.3E-10, 1.8E-4, 7.7E-5]
- Sensitivity to simultaneous change in feed rate and upper bound on reactor conversion

Multi-Scenario Optimization



1. Design plant to deal with different operating scenarios (over time or with uncertainty)

2. Can solve overall problem simultaneously
   - large and expensive
   - polynomial increase with number of cases
   - must be made efficient through specialized decomposition

3. Solve also each case independently as an optimization problem (inner problem with fixed design)
   - overall coordination step (outer optimization problem for design)
   - require sensitivity from each inner optimization case with design variables as external parameters

Design Under Uncertain Model Parameters and Variable Inputs

$$\begin{aligned} \min\ & E_{\theta\in\Theta}[P(d,z,y,\theta), \\ s.t.\ & h(d,z,y,\theta)=0, \\ & g(d,z,y,\theta)\le 0] \end{aligned}$$

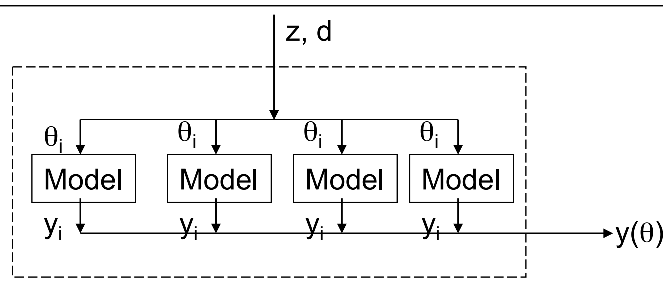
E[P, …] : expected value of an objective function
h : process model equations
g : process model inequalities
y : state variables (x, T, p, etc)
d : design variables (equipment sizes, etc)
θ_p : uncertain model parameters
θ_v : variable inputs **$\theta = [\theta_p^T\ \theta_v^T]$**
z : control/operating variables (actuators, flows, etc)
(may be fixed or a function of (some) θ)
(single or two stage formulations)

Multi-scenario Models for Uncertainty

$$\underset{d,z}{Min}\; E_{\theta\in\Theta}[P(d,z,y,\theta),$$
$$s.t.\; h(d,z,y,\theta)=0,$$
$$g(d,z,y,\theta)\le 0]$$

Some References: Bandoni, Romagnoli and coworkers (1993-1997), Narraway, Perkins and Barton (1991), Srinivasan, Bonvin, Visser and Palanki (2002), Walsh and Perkins (1994, 1996)

Multi-scenario Models for Uncertainty



$$Min\; f_0(d)+\sum_j \omega_j f_j(d,z,y_j,\theta_j)$$
$$s.t.\, h_j(d,z,y_j,\theta_j)=0$$
$$g_j(d,z,y_j,\theta_j)\le 0$$

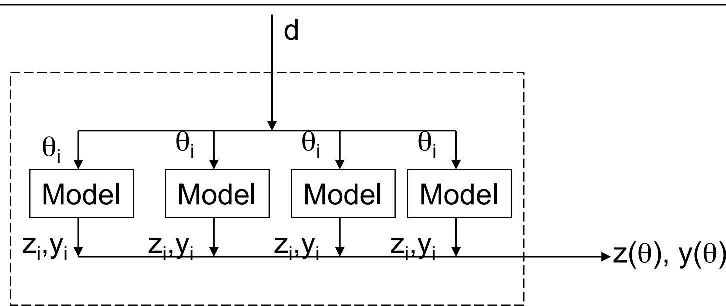
Some References: Bandoni, Romagnoli and coworkers (1993-1997), Narraway, Perkins and Barton (1991), Srinivasan, Bonvin, Visser and Palanki (2002), Walsh and Perkins (1994, 1996)

Multi-scenario Models for Variability

$$\underset{d,z(\theta)}{Min}\; E_{\theta\in\Theta}[P(d,z(\theta),y,\theta),$$
$$s.t.\; h(d,z(\theta),y,\theta)=0,$$
$$g(d,z(\theta),y,\theta)\le 0]$$

Some References: Grossmann and coworkers (1983-1991), Ierapetritou, Acevedo and Pistikopoulos (1996), Pistikopoulos and coworkers (1995-2001)

Multi-scenario Models for Variability



$$Min\; f_0(d)+\sum_j \omega_j f_j(d,z_j,y_j,\theta_j)$$
$$s.t.\, h_j(d,z_j,y_j,\theta_j)=0$$
$$g_j(d,z_j,y_j,\theta_j)\le 0$$

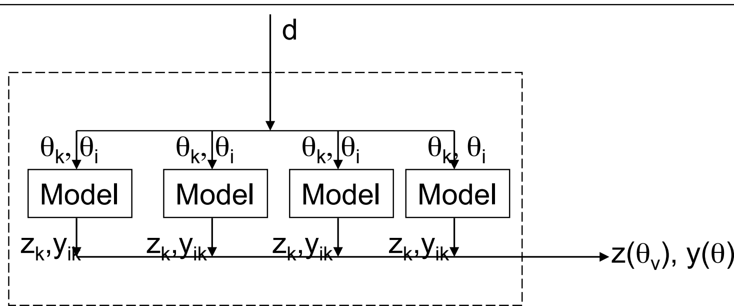
Some References: Grossmann and coworkers (1983-1991), Ierapetritou, Acevedo and Pistikopoulos (1996), Pistikopoulos and coworkers (1995-2001)

Multi-scenario Models for Both

$$\underset{\mathrm{d},z(\theta_v)}{Min}\; E_{\theta\in\Theta}[P(d,z(\theta_v),y,\theta),$$
$$s.t.\;\; h(d,z(\theta_v),y,\theta)=0,$$
$$g(d,z(\theta_v),y,\theta)\le 0]$$

Some References: Ostrovsky, Volin, Achenie (2003), Rooney, B. (2003)

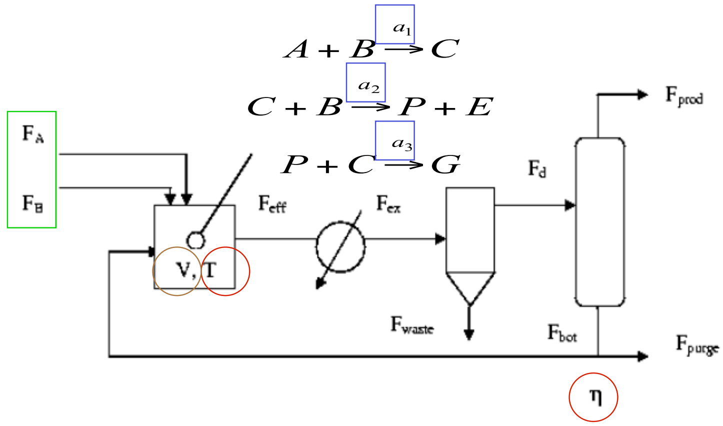
Multi-scenario Models for Both



$$Min\; f_0(d)+\sum_{i,k}\omega_{ik} f_{ik}(d,z_k,y_{ik},\theta_{v,k},\theta_{p,i})$$
$$s.t.\; h_{ik}(d,z_k,y_{ik},\theta_{v,k},\theta_{p,i})=0$$
$$g_{ik}(d,z_k,y_{ik},\theta_{v,k},\theta_{p,i})=0$$

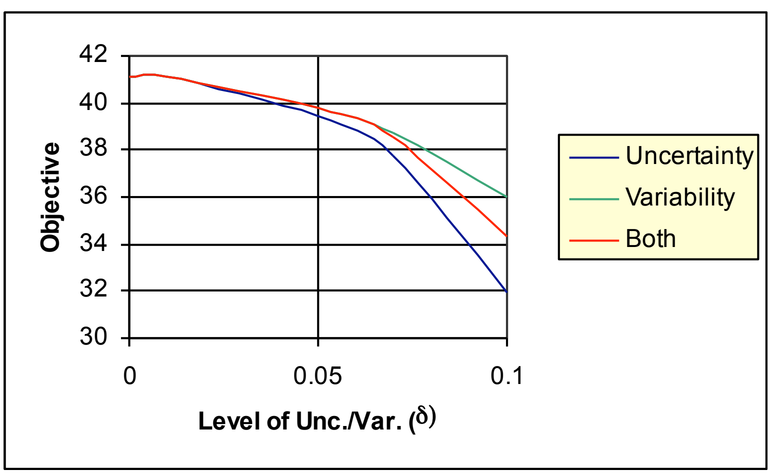
Some References: Ostrovsky, Volin, Achenie (2003), Rooney, B. (2003)

Example: Williams-Otto Process
(Rooney, B., 2003)



- Uncertain model parameters, a_1, a_2 and a_3
- Varying process parameters: $F_A = 10000(1 \pm \delta)$ and $F_B = 40000(1 \pm \delta)$

Uncertainty and Variability: Williams-Otto Process
(Rooney, B., 2003)



- Uncertain model parameters, a_1, a_2 and a_3
- Varying process parameters: $F_A = 10000(1 \pm \delta)$ and $F_B = 40000(1 \pm \delta)$

Solving Multi-scenario Problems: Interior Point Method

$$Min\ f_0(d) + \sum_j \omega_j f_j(d, z_j, y_j, \theta_j)$$
$$s.t.\ h_j(d, z_j, y_j, \theta_j) = 0$$
$$g_j(d, z_j, y_j, \theta_j) + s_j = 0, s_j \geq 0$$

$$Min\ f_0(p) + \sum_j \omega_j f_j(p, x_j)$$
$$s.t.\ c_j(p, x_j) = 0,\ \ p, x_j \geq 0$$

$$Min\ f_0(p) + \sum_j \omega_j f_j(p, x_j) - \mu \left\{ \sum_{j,l} \ln x_j^l + \sum_{j,l} \ln p^l \right\}$$
$$s.t.\ c_j(p, x_j) = 0$$

$$\mu^i \rightarrow 0 \Rightarrow [x(\mu^i), p(\mu^i)] \rightarrow [x^*, p^*]$$

Newton Step for IPOPT

$$\begin{bmatrix} \boxed{K_1} & & & & & \boxed{w_1} \\ & \boxed{K_2} & & & & w_2 \\ & & \boxed{K_3} & & & w_3 \\ & & & \ldots & & \ldots \\ & & & & \boxed{K_N} & w_N \\ w_1^T & w_2^T & w_3^T & \ldots & w_N^T & K_p \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \ldots \\ u_N \\ u_p \end{bmatrix} = - \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ \ldots \\ r_N \\ r_p \end{bmatrix}$$

$$K_i = \begin{bmatrix} (\nabla_{x_i, x_i} L^k + (X_i^k)^{-1} V_i^k) & \nabla_{x_i} c_i(x_i^k, p^k) \\ \nabla_{x_i} c_i(x_i^k, p^k)^T & 0 \end{bmatrix} \quad u_i = \begin{bmatrix} \Delta x_i \\ \Delta \lambda_i \end{bmatrix} \quad u_p = \begin{bmatrix} \Delta p \\ \Delta \bar{\lambda} \end{bmatrix}$$

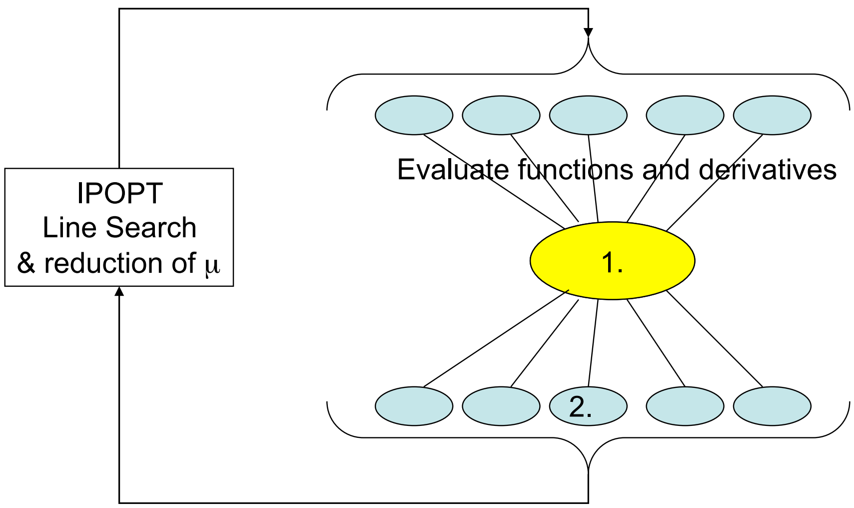
$$K_p = \begin{bmatrix} \nabla_{p,p} L^k + (P^k)^{-1} V_p^k & \nabla_p \bar{c} \\ \nabla_p \bar{c}^T & 0 \end{bmatrix} \quad w_i = \begin{bmatrix} \nabla_{x_i p} L & \nabla_{x_i} \bar{c} \\ \nabla_p c_i^T & \end{bmatrix}$$

Schur Complement Decomposition Algorithm

Key Steps

$$1. \left(K_{pp} - \sum_i w_i^T K_i^{-1} w_i \right) \Delta u_p = r_p - \sum_i w_i^T K_i^{-1} r_i$$

$$2. \; K_i \Delta u_i = r_i - w_i \Delta u_p$$



Computational cost is linear in number of periods
Trivial to parallelize

Nonlinear Optimization Engines

Evolution of NLP Solvers:

➔ ***process optimization for design, control and operations***

SQP → rSQP → IPOPT → IPOPT 3.x

'00s: Simultaneous dynamic optimization over 1 000 000 variables and constraints

Object Oriented Codes to tailor structure, architecture to problems

Multi-scenario Design Model

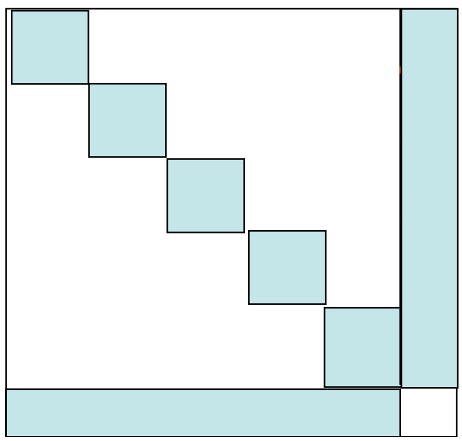
$$Min\ f_0(d) + \Sigma_i f_i(d, x_i)$$
$$s.t.\ h_i(x_i, d) = 0,\ i = 1,\dots N$$
$$g_i(x_i, d) \le 0,\ i = 1,\dots N$$
$$r(d) \le 0$$

Variables:

x: state (z) and control (u) variables in each operating period

d: design variables (e. g. equipment parameters) used

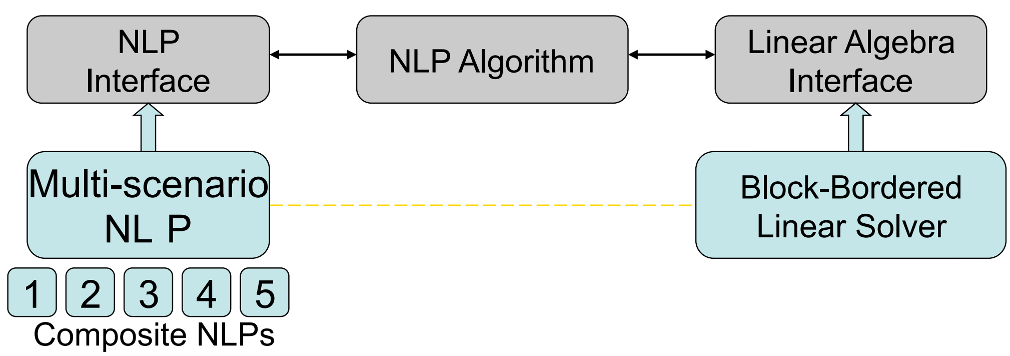
δ_i: substitute for d in each period and add $\delta_i = d$

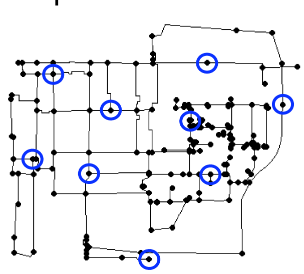


Composite NLP

$$Min\ \ \Sigma_i\ (f_i(\delta_i, x_i) + f_0(\delta_i)/N)$$
$$s.t.\ h_i(x_i, \delta_i) = 0,\ i = 1,\dots N$$
$$g_i(x_i, \delta_i) + s_i = 0,\ i = 1,\dots N$$
$$0 \le s_i,\ \underline{d - \delta_i = 0},\ i = 1,\dots N$$
$$r(d) \le 0$$

Internal Decomposition Implementation





- Water Network Base Problem
  - 36,000 variables
  - 600 common variables
- Testing
  - Vary # of scenarios
  - Vary # of common variables

Parallel Schur-Complement Scalability

Multi-scenario Optimization

- Single Optimization over many scenarios, performed on parallel cluster

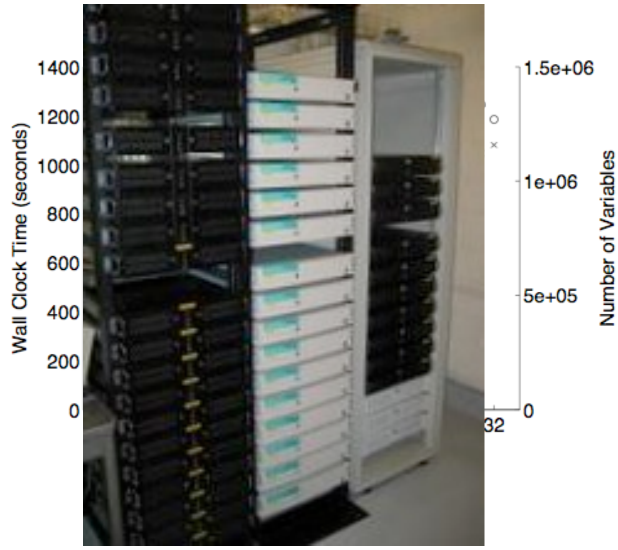
Water Network Case Study

- 1 basic model
  - Nominal design optimization
- 32 possible uncertainty scenarios
  - Form individual blocks

Determine Injection time profiles as common variables

Characteristics

- 36,000 variables per scenario
- 600 common variables



Summary and Conclusions

Optimization Algorithms
-Unconstrained Newton and Quasi Newton Methods
-KKT Conditions and Specialized Methods
-Reduced Gradient Methods (GRG2, MINOS)
-Successive Quadratic Programming (SQP)
-Reduced Hessian SQP
-Interior Point NLP (IPOPT)

Process Optimization Applications
-Modular Flowsheet Optimization
-Equation Oriented Models and Optimization
-Realtime Process Optimization
-Blending with many degrees of freedom

Further Applications
-Sensitivity Analysis for NLP Solutions
-Multi-Scenario Optimization Problems