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

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

Large-Scale Nonlinear Programming
- Data Reconciliation
- Real-time Process Optimization

Further Applications
- Sensitivity Analysis for NLP Solutions
- Multiperiod Optimization Problems

Summary and Conclusions

Introduction

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

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

<u>Decision Variables</u>: 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, 1998

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

$$
\begin{aligned}
&\min \; f(x, y) \\
&st \; h(x, y) = 0 \\
&\quad g(x, y) \leq 0 \\
&x \in R^n \quad y \in \{0, 1\}^{ny}
\end{aligned}
$$

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?

<u>Constrained Problem</u>

$$\text{Min}\left\{C_T \frac{\pi D^2}{2} + C_S \pi DL = \text{cost}\right\} \qquad (1)$$

$$\text{s.t.} \qquad V - \frac{\pi D^2 L}{4} = 0$$

<u>Convert to Unconstrained (</u>Eliminate L)

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

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

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

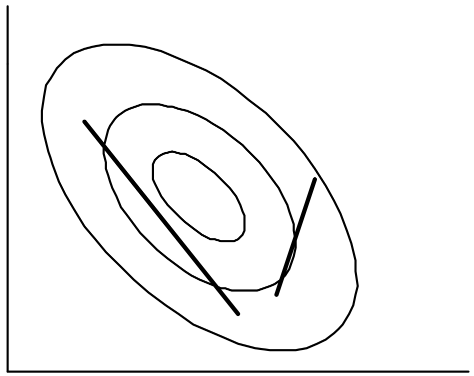
$$\Longrightarrow L/D = C_T/C_S$$

Note:

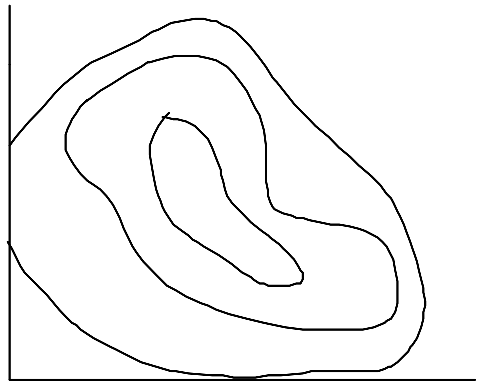
- What if L cannot be eliminated in (1) explicitly? (strange shape)
- What if D cannot be extracted from (2)? (cost correlation implicit)

Two Dimensional Contours of F(x)

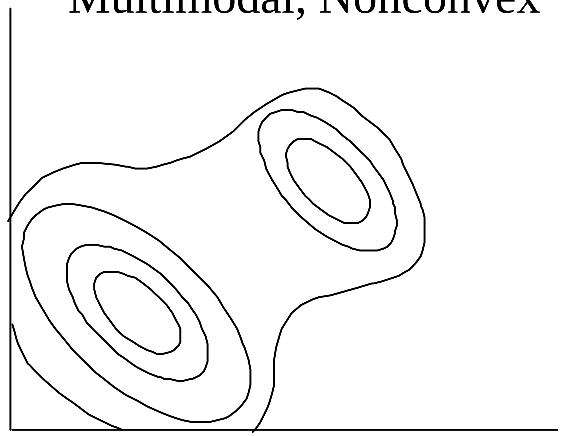
Convex Function

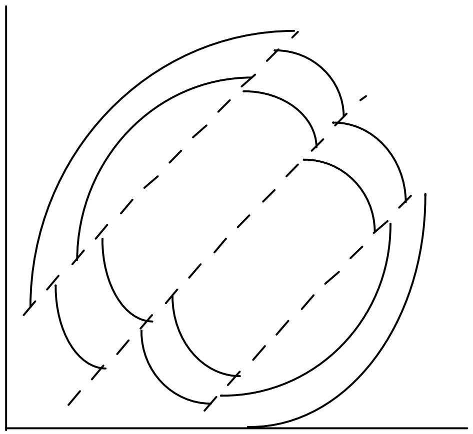


Nonconvex Function

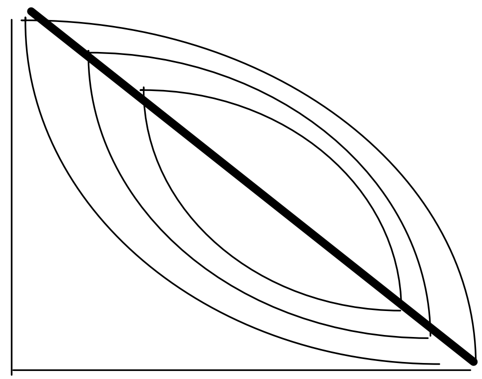


Multimodal, Nonconvex





Discontinuous



Nondifferentiable (convex)

Local vs. Global Solutions

- Find a *local minimum* point x* for f(x) for feasible region defined by constraint functions: *f(x*) ≤ f(x) for all x satisfying the constraints in some neighborhood around x** <u>*(not for all x* $\in$ **X***)*</u>
- Finding and verifying *global solutions* will not be considered here.
- Requires a more expensive search (e.g. spatial branch and bound).
- A local solution to the NLP is also a global solution under the following *sufficient* conditions based on convexity.
- *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**.

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 - ($\nabla f(x)$)

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

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

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

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

  $\det(A\,B) = \det(A)\det(B)$; $\det(A) = \det(A^T)$

  $\det(\alpha A) = \alpha^n \det(A)$; $\det(A) = \Pi_i\, \lambda_i(A)$

- 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, $\sigma = \lambda(A^TA)^{1/2} \geq 0$

- 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)

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

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: $Q(x) = a^T x + ½ x^T A x$

Define: $z = V^T x$ and $Q(Vz) = (a^T V) z + ½ z^T (V^T A V) z$

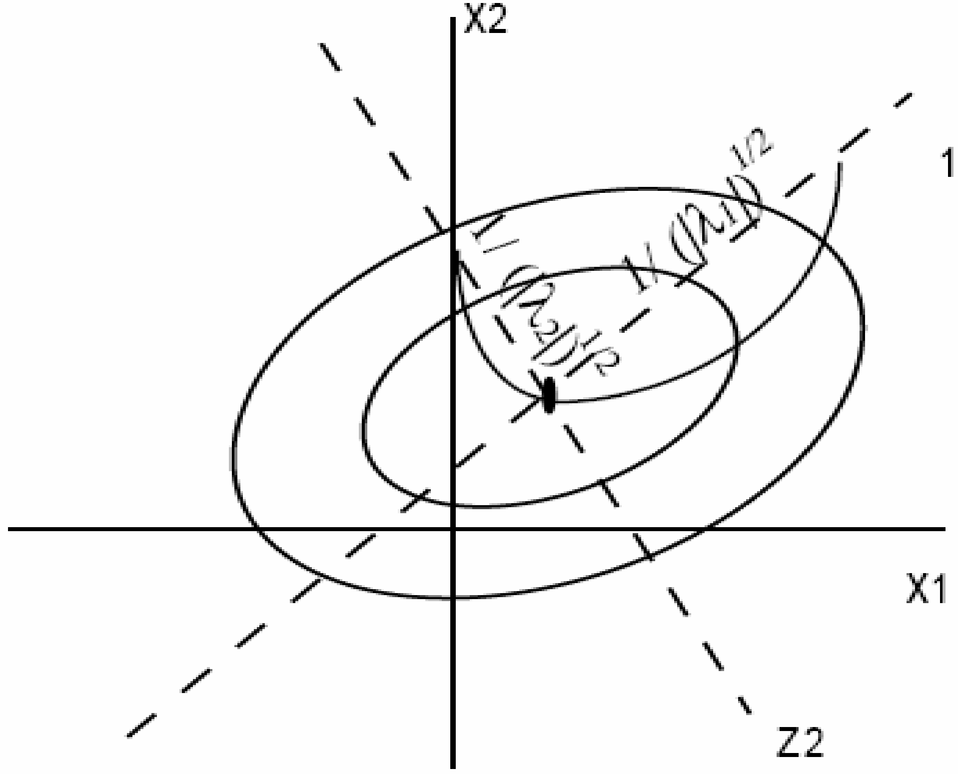
$$= (a^T V) z + ½ z^T \Lambda z$$

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

Positive (or Negative) Curvature
Positive (or Negative) Definite Hessian

Both eigenvalues are strictly positive or negative

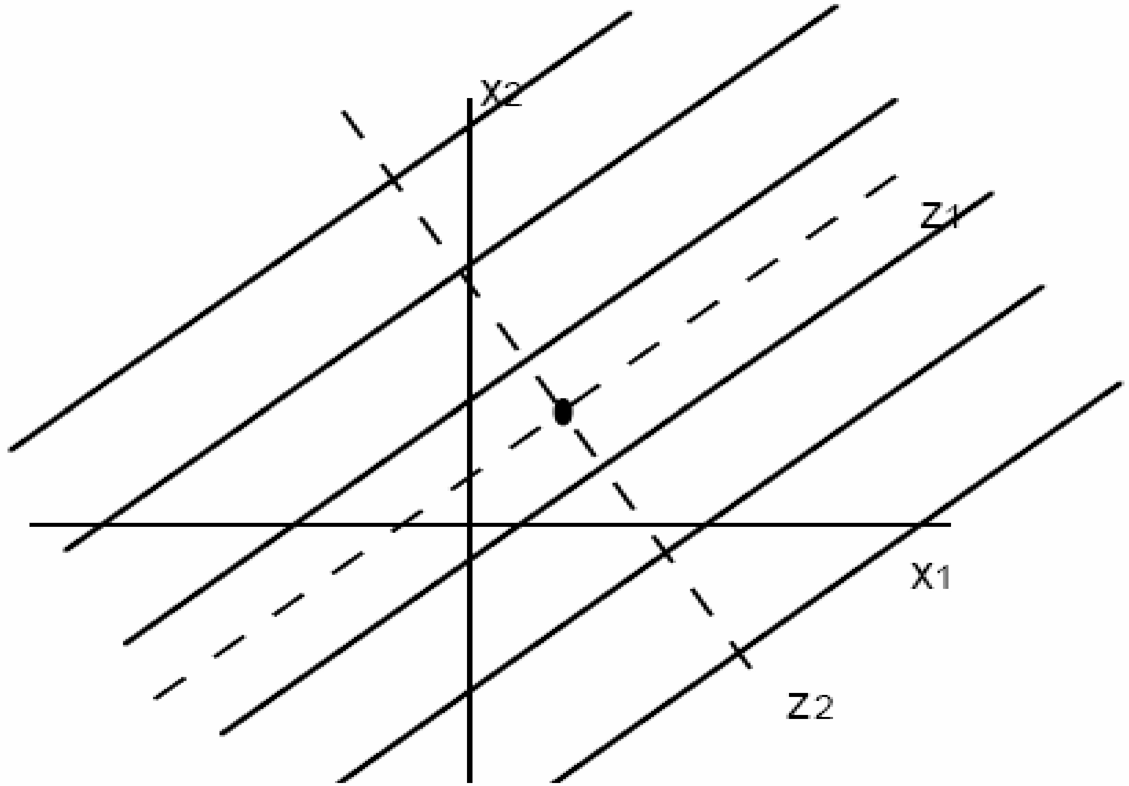
- A is positive definite or negative definite
- Stationary points are minima or maxima



Zero Curvature
Singular Hessian

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

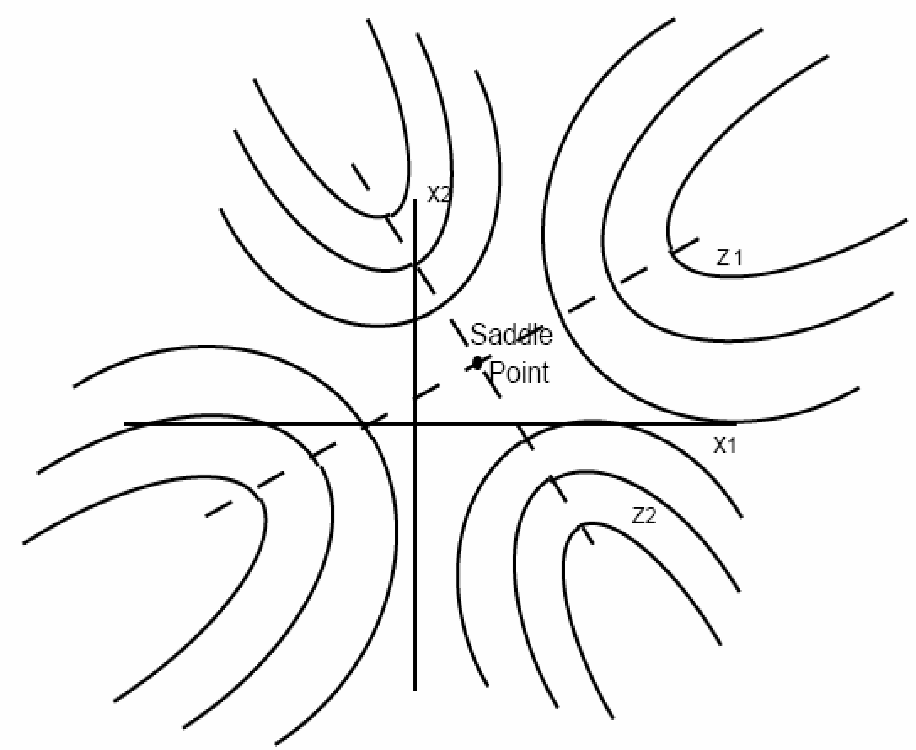
- A is positive semidefinite or negative semidefinite
- There is a ridge of stationary points (minima or maxima)



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

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

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

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

- **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

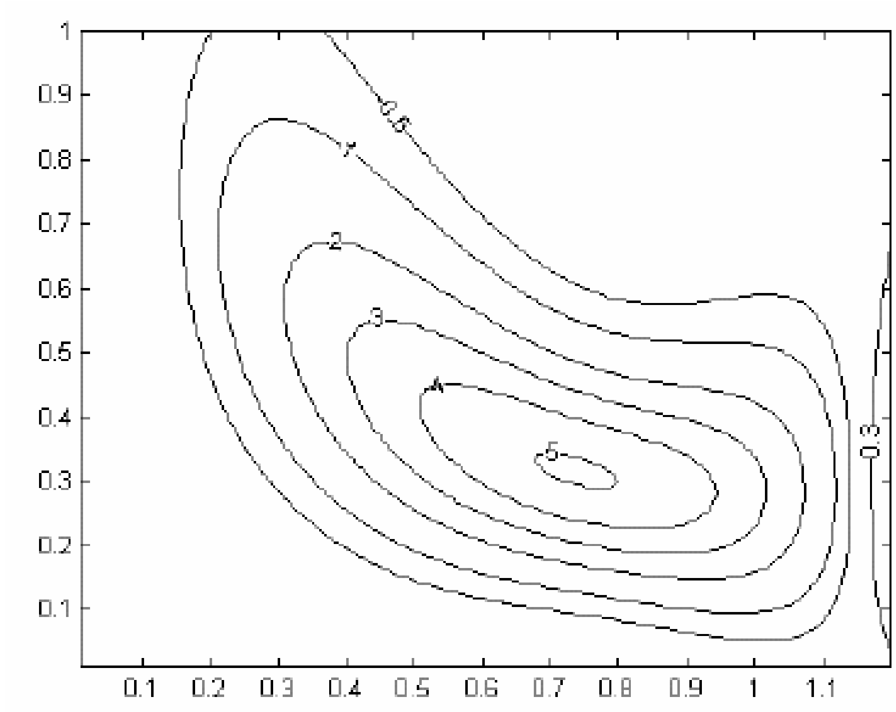
- Global Convergence - will it converge to a local optimum (or stationary point) from a poor starting point?
- Local Convergence Rate - how fast will it converge close to the solution?

2. Benchmarks on Large Class of Test Problems

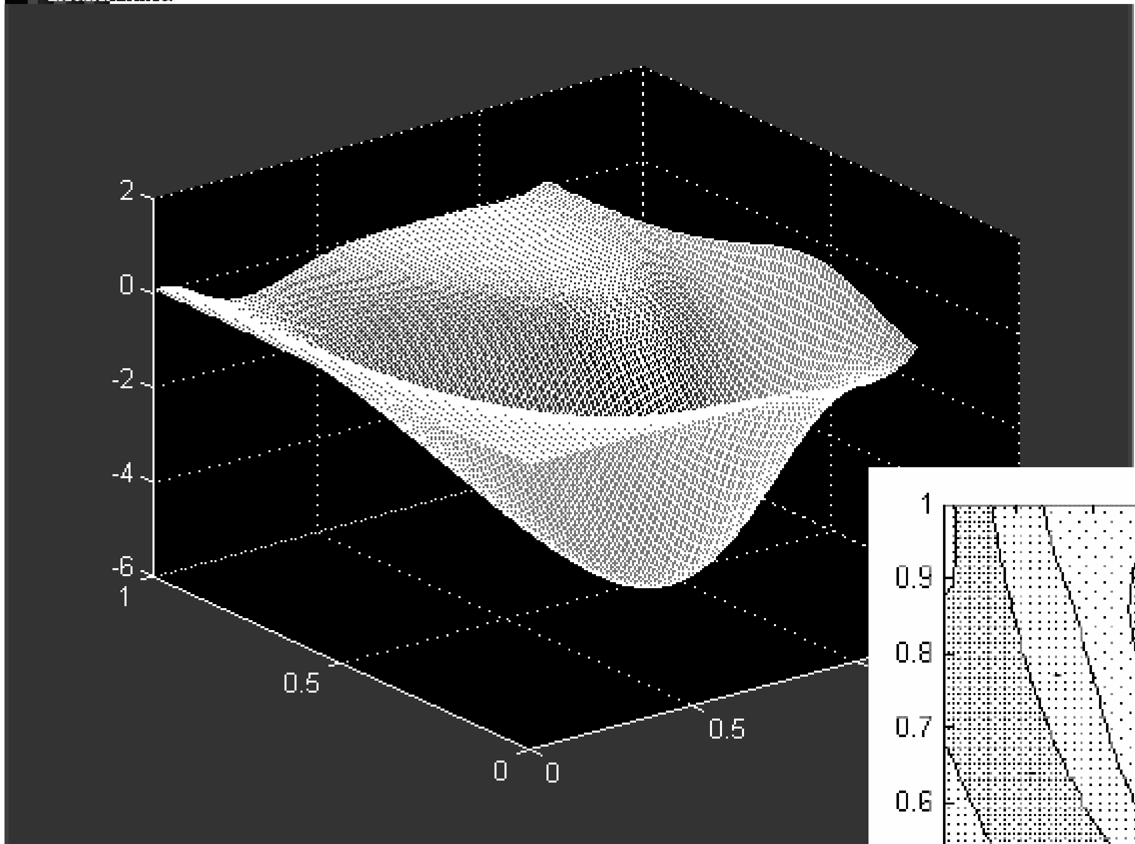
Representative Problem (Hughes, 1981)

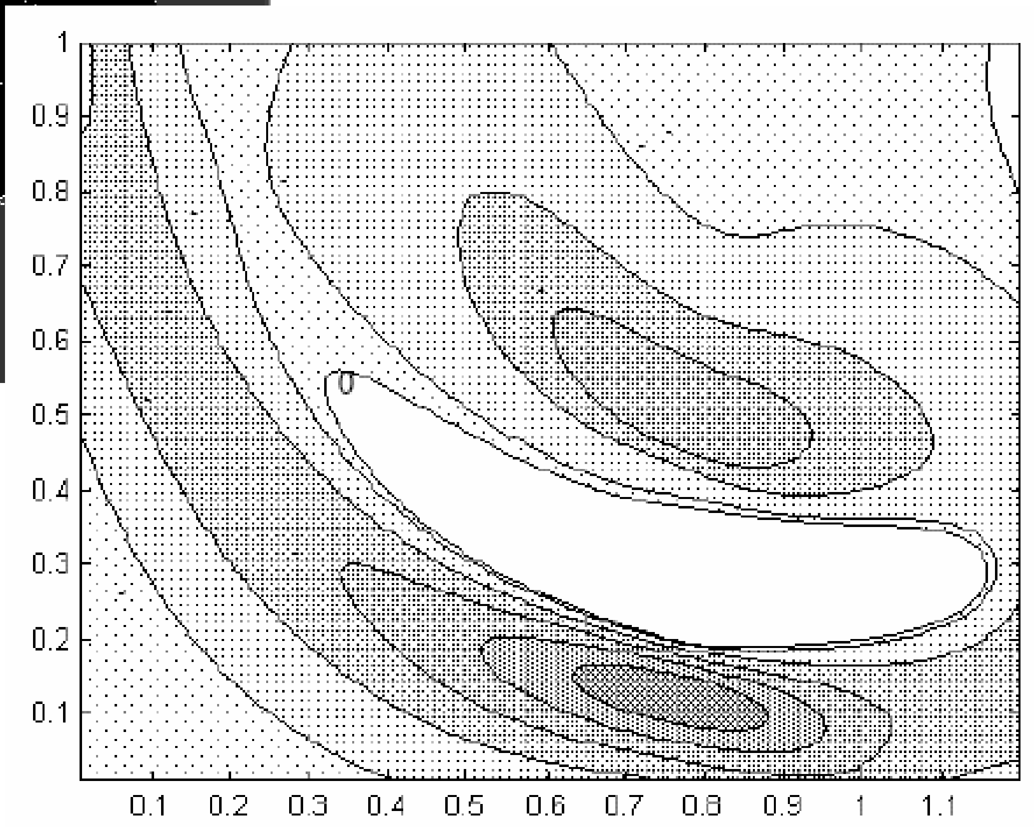
$$\begin{aligned}
&Min\ \ f(x_1, x_2) = \alpha\, exp(-\beta) \\
&u = x_1 - 0.8 \\
&v = x_2 - (a_1 + a_2 u^2 (1- u)^{1/2} - a_3 u) \\
&\alpha = -b_1 + b_2 u^2 (1+u)^{1/2} + b_3 u \\
&\beta = c_1 v^2 (1 - c_2 v)/(1+ c_3 u^2)
\end{aligned}$$

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



Three Dimensional Surface and Curvature for Representative Test Problem

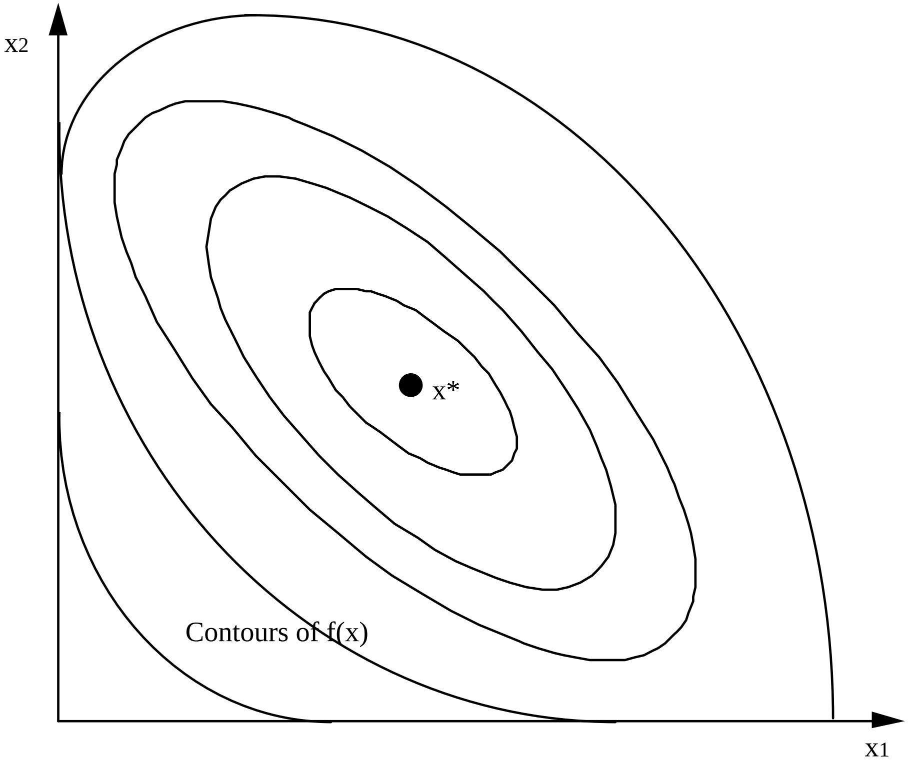




Regions where minimum eigenvalue is less than:

[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*:*

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

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

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

- $f(x)$ 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:

$$\| x^{k+1} - x^* \| = K \| x^k - x^* \|^2$$

Basic Newton Algorithm - Line Search

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

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

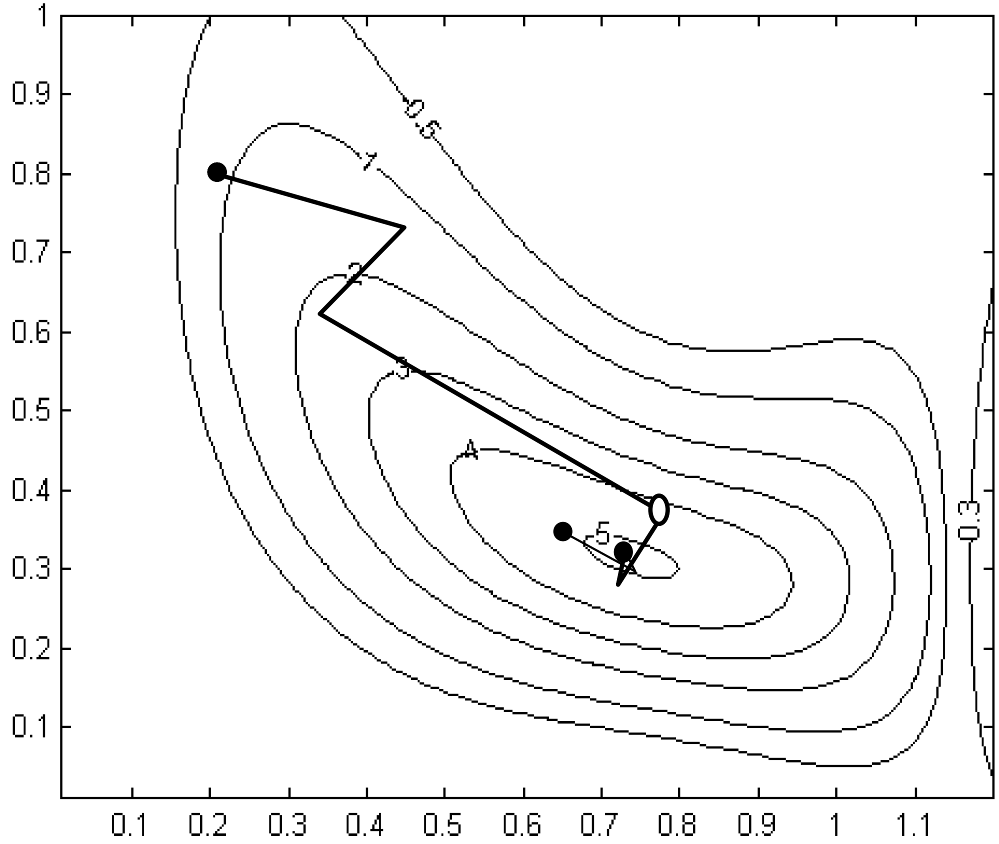
2. Evaluate $B^k = \nabla^2 f(x^k)$ or an approximation.

3. Solve: $\mathbf{B^k\, d = -\nabla f(x^k)}$
   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



<u>Starting Points</u>

[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.

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

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

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

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:

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

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

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

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

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

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

Quasi-Newton Methods

<u>BFGS Formula</u> (Broyden, Fletcher, Goldfarb, Shanno, 1970-71)

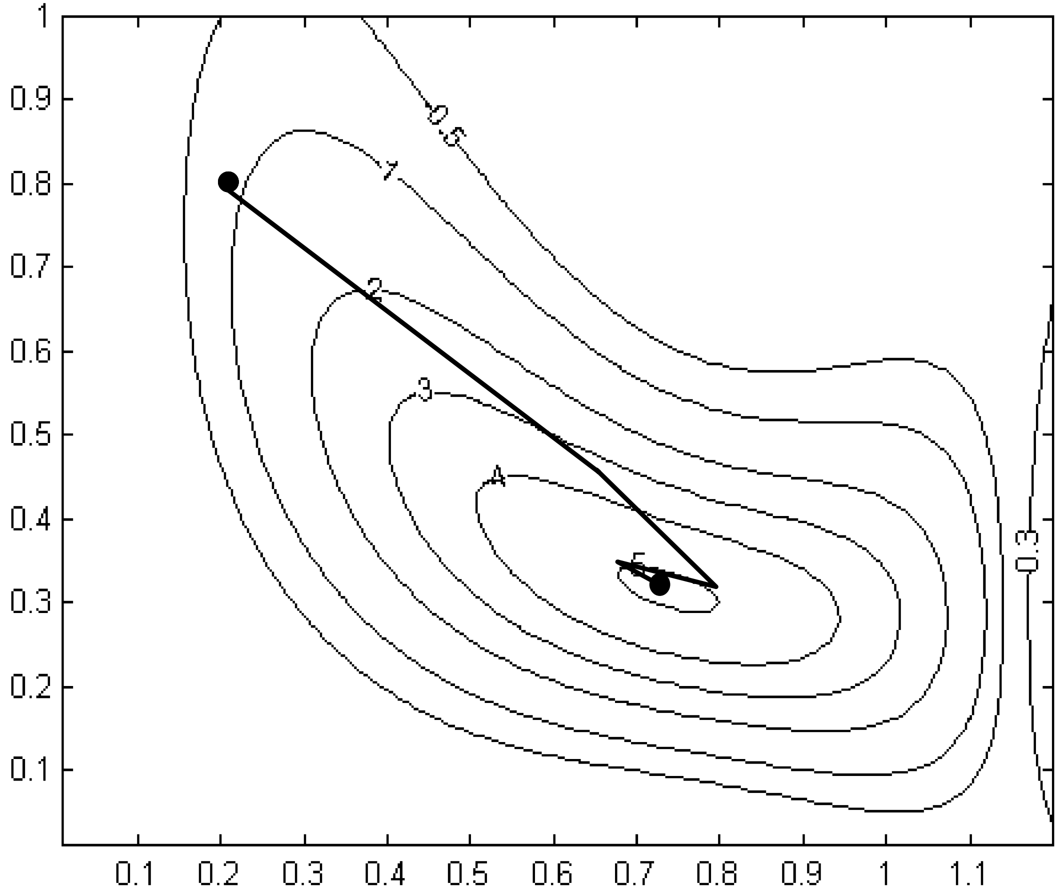
$$B^{k+1} = B^k + \frac{y y^T}{s^T y} - \frac{B^k s\, s^T B^k}{s\, B^k s}$$

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

<u>Notes:</u>

1) Both formulas are derived under <u>similar</u> <u>assumptions</u> and have symmetry
2) Both have <u>superlinear</u> <u>convergence</u> 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 <u>best</u> methods for general purpose problems if second derivatives are not available.

Quasi-Newton Method - BFGS Convergence Path



Starting Point

[0.8, 0.2] 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: $\qquad Min_x \; f(x)$

$\qquad$ *s.t.* $\qquad g(x) \leq 0$

$\qquad\qquad\qquad h(x) = 0$

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 Unique Optimum

- $f(x)$ must be *convex*, and
- feasible region must be convex,

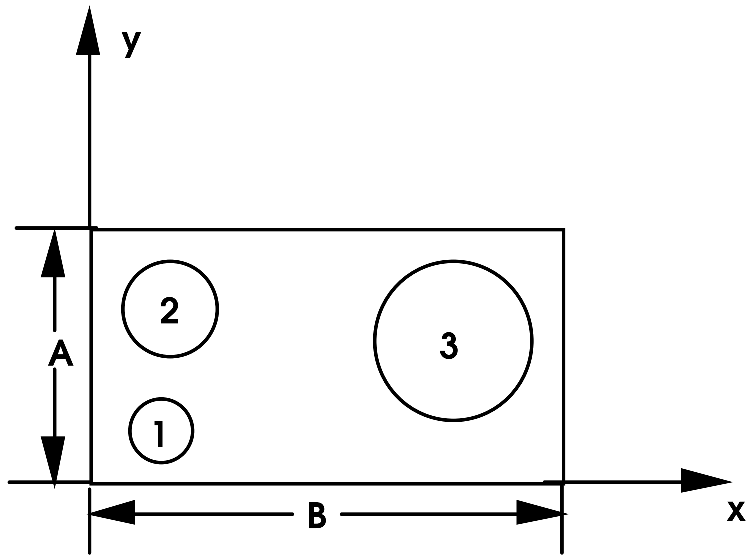
i.e. $g(x)$ are all *convex*

$h(x)$ are all *linear*

Except in special cases, ther 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.



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

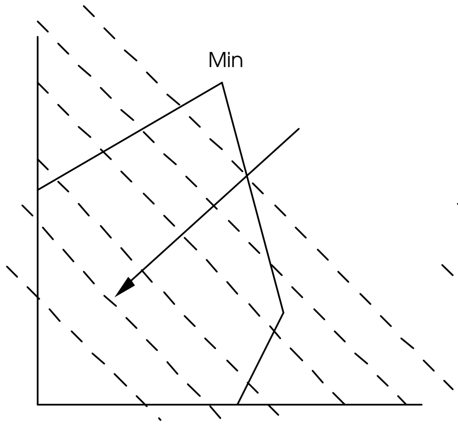
in box

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

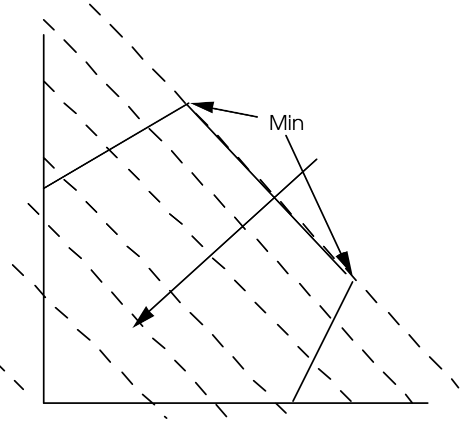
$$\begin{cases} (x_1 - x_2)^2 + (y_1 - y_2)^2 \ge (R_1 + R_2)^2 \\ (x_1 - x_3)^2 + (y_1 - y_3)^2 \ge (R_1 + R_3)^2 \\ (x_2 - x_3)^2 + (y_2 - y_3)^2 \ge (R_2 + R_3)^2 \end{cases}$$

no overlaps

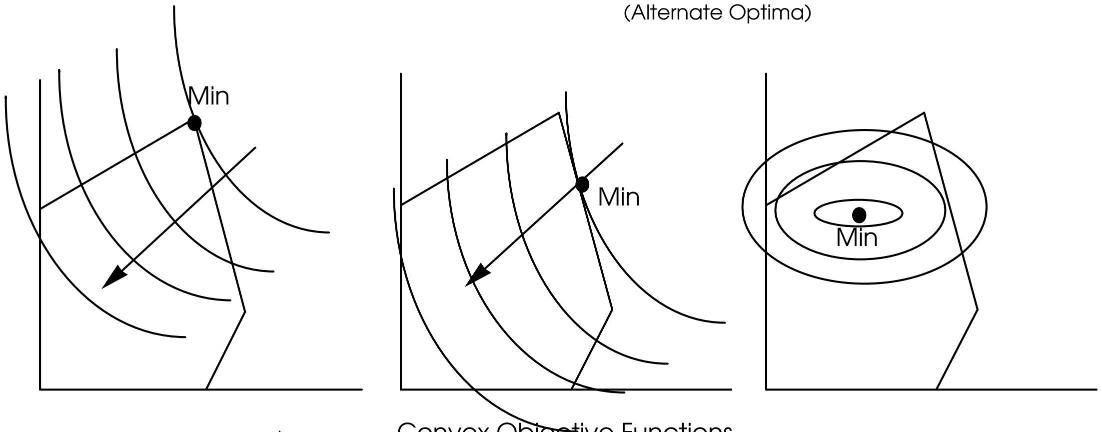
Characterization of Constrained Optima



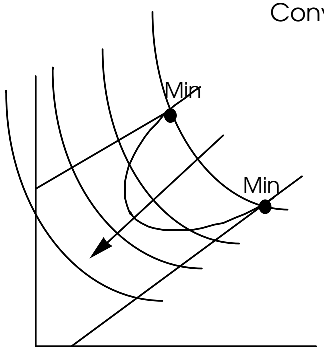
Linear Program



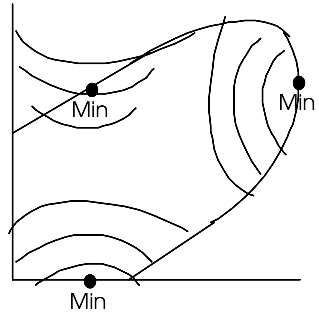
Linear Program
(Alternate Optima)



Convex Objective Functions
Linear Constraints

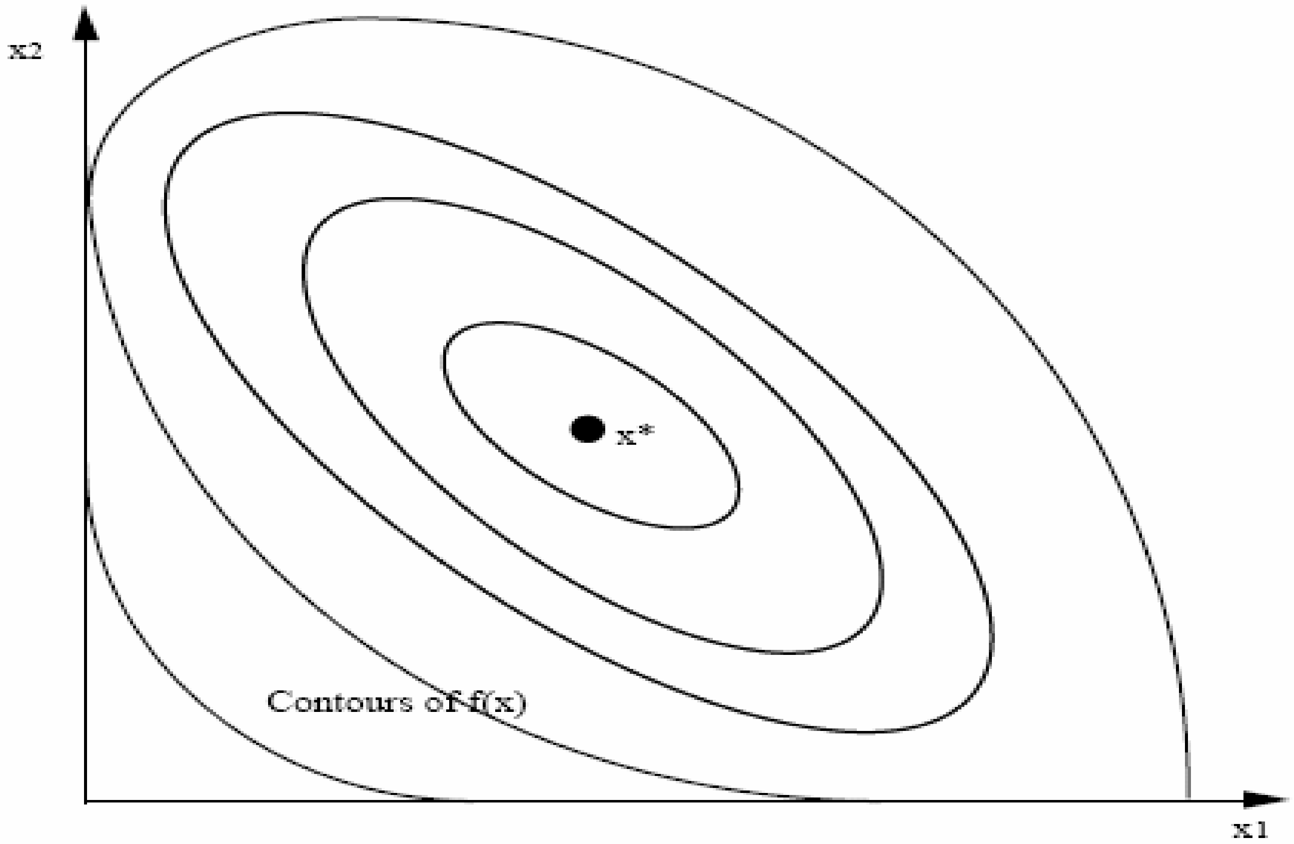


Nonconvex Region
Multiple Optima



Nonconvex Objective
Multiple 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)

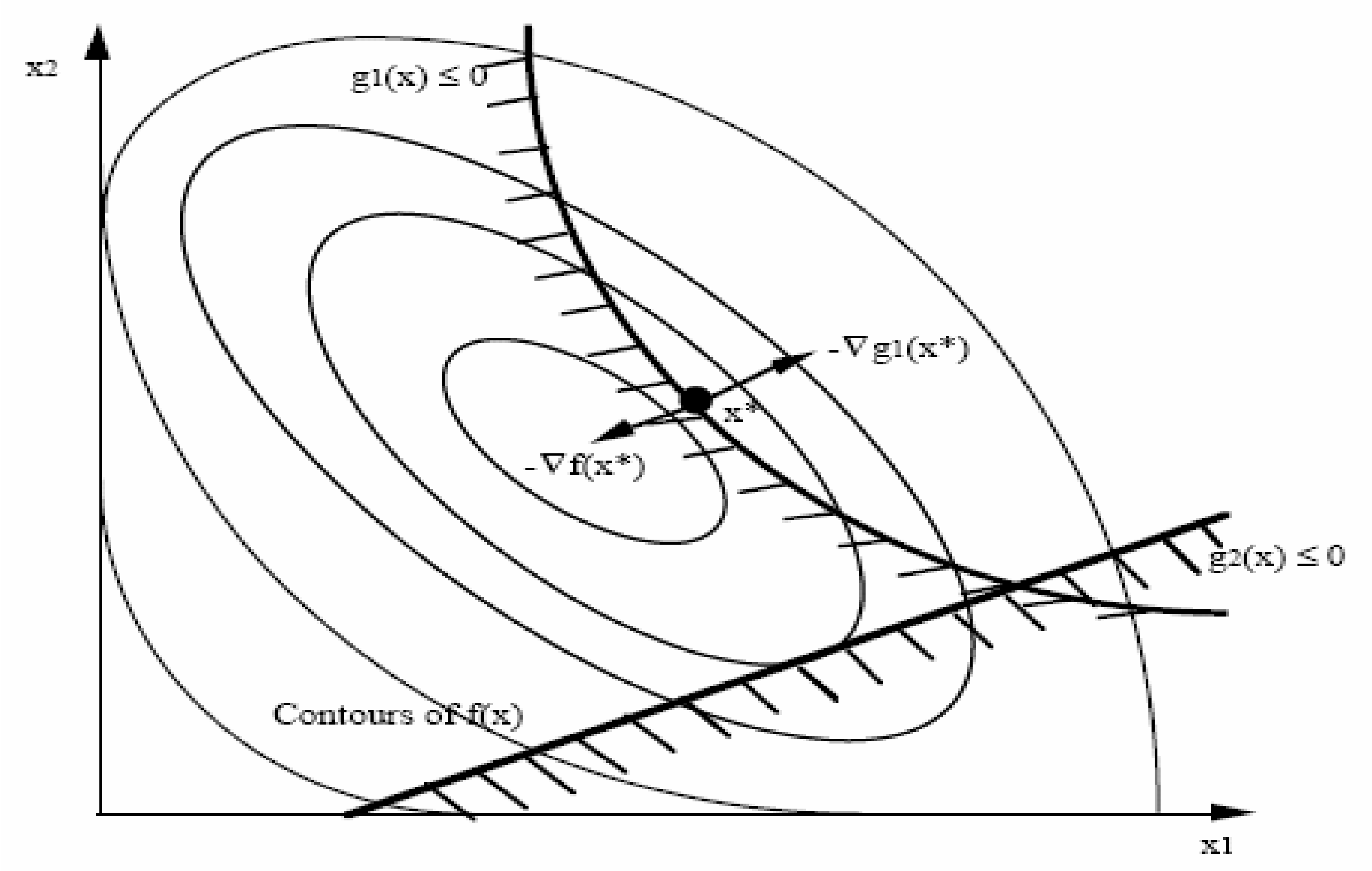
<u>Unconstrained Local Minimum</u>
<u>Sufficient Conditions</u>

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

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

(positive definite)

Optimal solution for inequality constrained problem



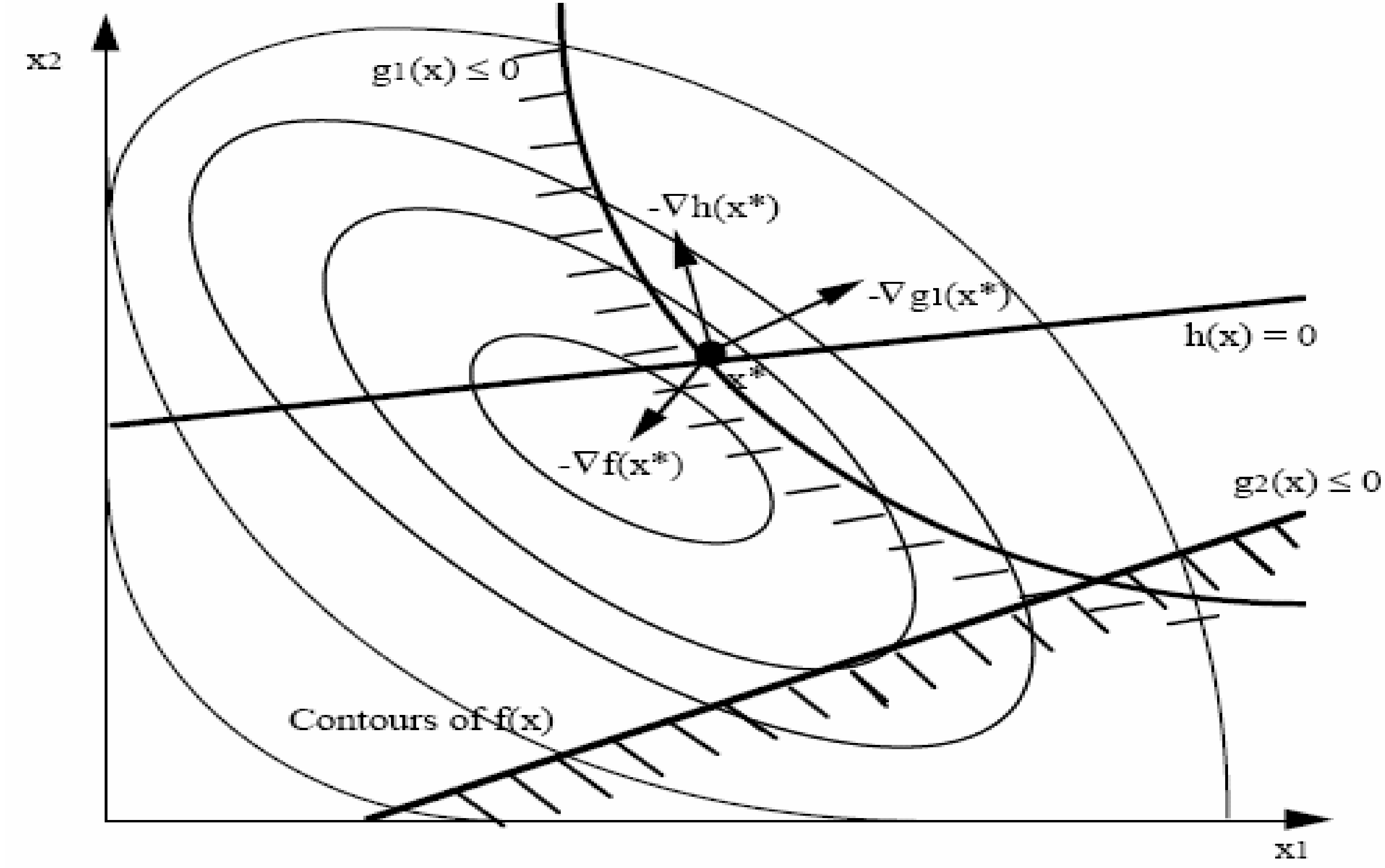
Min $f(x)$

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

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: ∇f, ∇g_1, ∇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

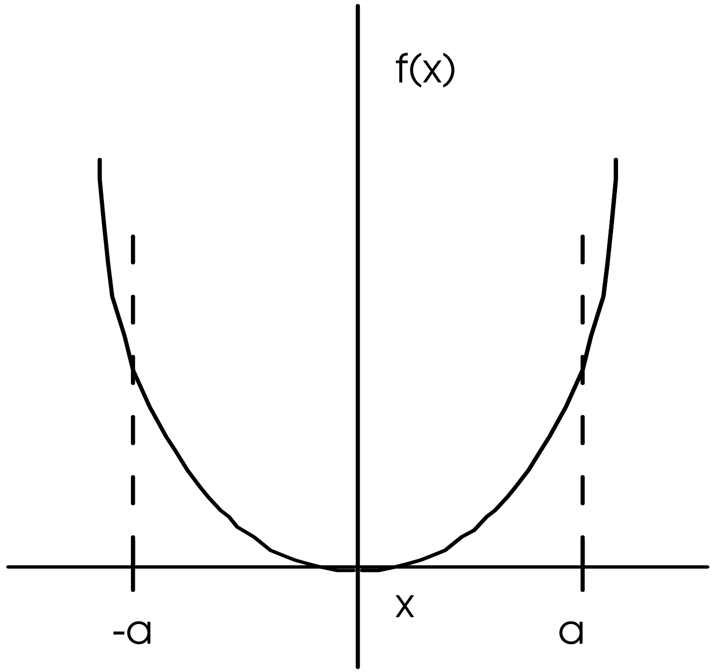
- Positive curvature in "constraint" directions.
- $p^T \nabla^2 L(x^*)\, p \geq 0$ $(p^T \nabla^2 L(x^*)\, p > 0)$
  where p are the constrained directions: $\nabla g_A(x^*)^T p = 0,\ \nabla h(x^*)^T p = 0$

Single Variable Example of KKT Conditions

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

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

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



Consider three cases:

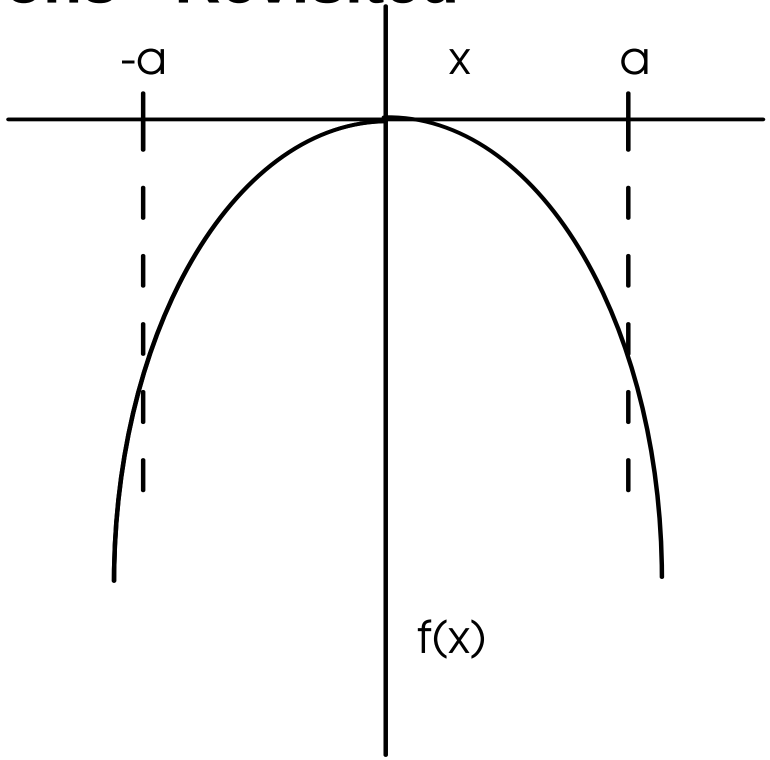
- $u_1 > 0,\ u_2 = 0$ Upper bound is active, $x = a,\ u_1 = -2a,\ u_2 = 0$
- $u_1 = 0,\ u_2 > 0$ Lower bound is active, $x = -a,\ u_2 = -2a,\ u_1 = 0$
- $u_1 = u_2 = 0$ Neither bound is active, $u_1 = 0,\ u_2 = 0,\ x = 0$

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:

- $u_1 > 0,\ u_2 = 0$ Upper bound is active, $x = a,\ u_1 = 2a,\ u_2 = 0$
- $u_1 = 0,\ u_2 > 0$ 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.

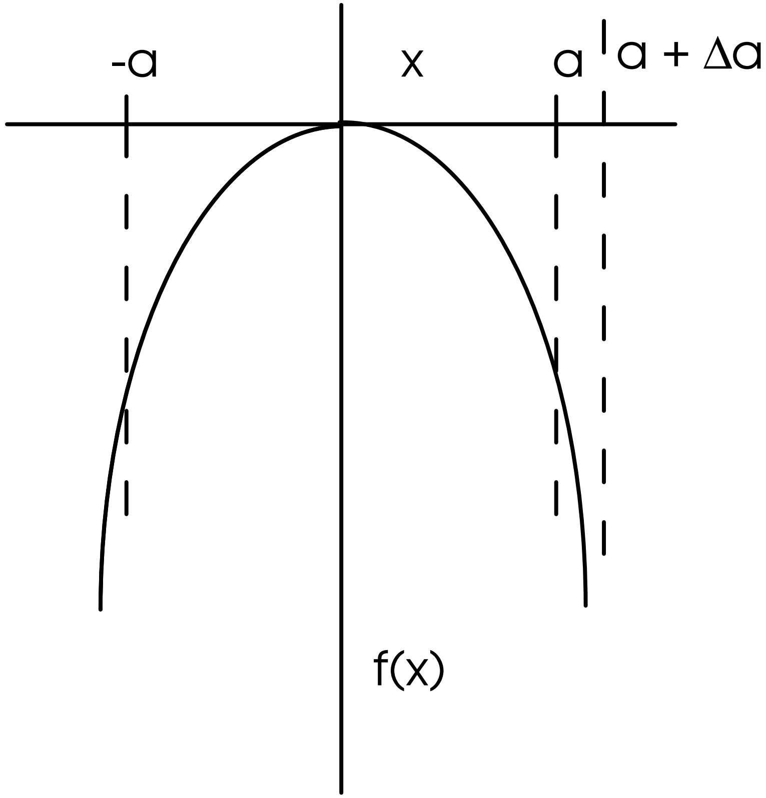
For this problem, however, there are no nonzero allowable directions that satisfy this condition. Consequently the solution x^* is defined entirely by the active constraint. The condition:

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

for all 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

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

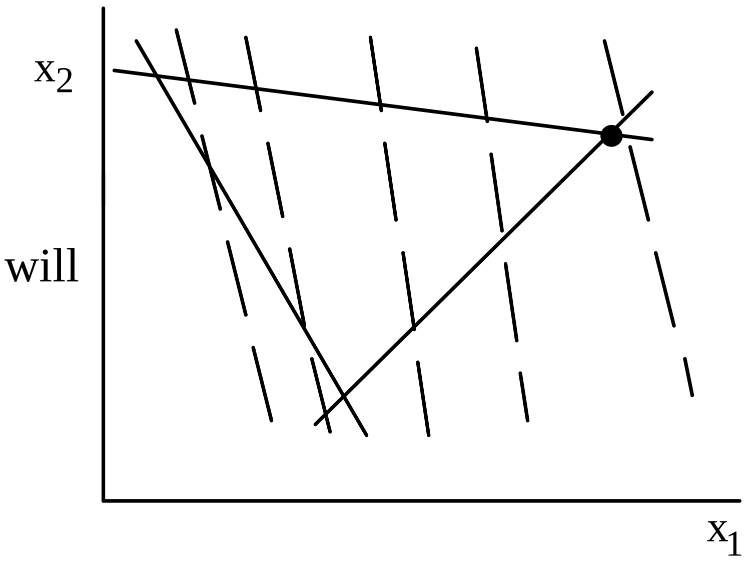
$$df(x^*)/da = 2a = u_1$$

Special Cases of Nonlinear Programming

Linear Programming:

$$\begin{aligned} \text{Min} \quad & c^T x \\ \text{s.t.} \quad & Ax \le b \\ & Cx = d, \ x \ge 0 \end{aligned}$$

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

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

Now, define $f = -2x_1 - 3x_2 \quad \Rightarrow \quad f + 2x_1 + 3x_2 = 0$

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

x_1	x_2	$x3$	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: $\quad \text{Min} \quad a^T x + 1/2\, x^T B\, x$

$$A\,x \le b$$
$$C\,x = d$$

1) Can be solved using LP-like techniques:

   (Wolfe, 1959)

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

$$\text{s.t.} \quad a + Bx + A^T u + C^T v = z_+ - z_-$$
$$Ax - b + s = 0$$
$$Cx - d = 0$$
$$s, z_+, z_- \ge 0$$
$$\{u_j\, s_j = 0\}$$

   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.

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

$$Max \sum_i \mu_i x_i$$

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

$$x_i \geq 0, \; etc.$$

Note: maximize average return, no accounting for risk.

Portfolio Planning Problem

Definition of Risk - fluctuation of ri(t) over investment (or past) time period.
To minimize risk, minimize variance about portfolio mean (risk averse).

Variance/Covariance Matrix, S

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

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

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 VE 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 VE  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:

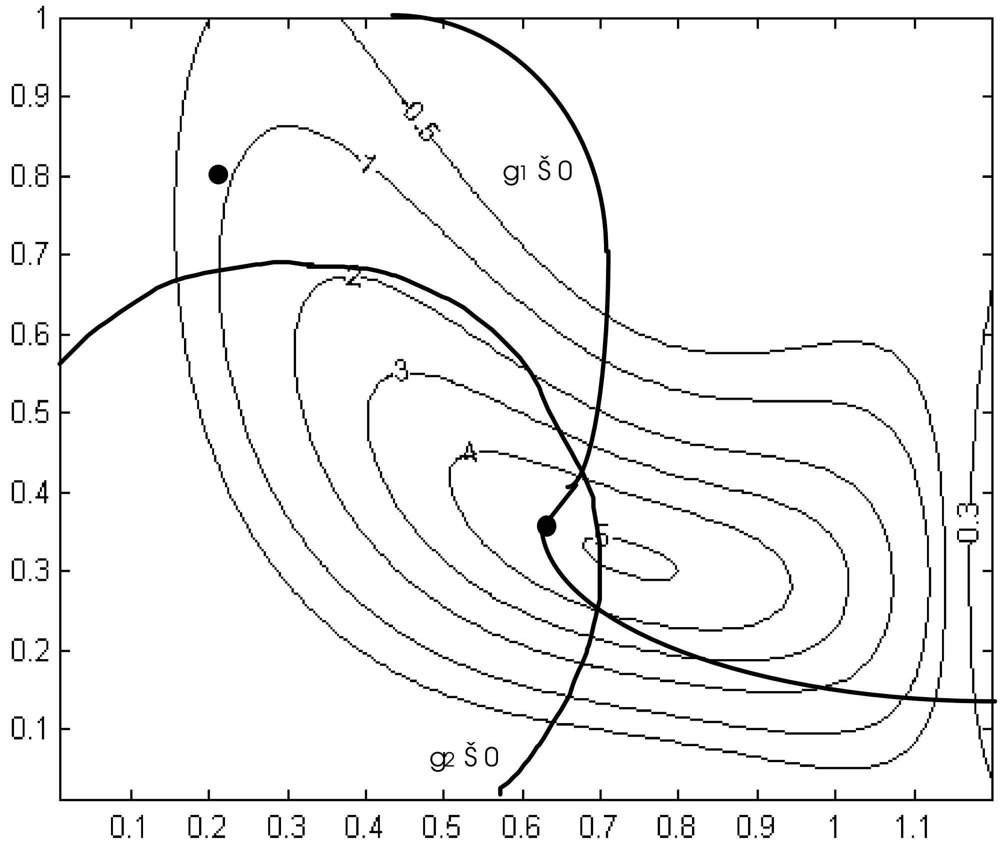
- Reduced Gradient Methods - (with Restoration) GRG2, CONOPT
- Reduced Gradient Methods - (without Restoration) MINOS
- Successive Quadratic Programming - generic implementations
- Penalty Functions - popular in 1970s, but fell into disfavor. Barrier Methods have been developed recently and are again 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 $f(x_1, x_2) = \alpha \exp(-\beta)$

$g1 = (x_2+0.1)^2[x_1^2+2(1-x_2)(1-2x_2)] - 0.16 \leq 0$

$g2 = (x_1 - 0.3)^2 + (x_2 - 0.3)^2 - 0.16 \leq 0$

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

Reduced Gradient Method with Restoration

(GRG2/CONOPT)

$Min \quad f(x)$

$s.t. \; g(x) + s = 0$ *(add slack variable)*

$c(x) = 0$

$a \le x \le b, \; s \ge 0$

$Min \quad f(z)$

‘$\Rightarrow \quad s.t. \; h(z) = 0$

$a \le z \le b$

- Partition variables into:

  z_B - dependent or basic variables

  z_N - nonbasic variables, fixed at a bound

  z_S - independent or superbasic variables

  Analogy to linear programming. Superbasics required only if nonlinear problem

- Solve unconstrained problem in space of superbasic variables.

  Let $z^T = [z_S^T \; z_B^T \, z_N^T]$ optimize wrt z_S with $h(z_S, z_B, z_N) = 0$

  Calculate *constrained derivative* or *reduced gradient* wrt z_S.

  •Dependent variables are $z_B \in R^m$

  •Nonbasic variables z_N (temporarily) fixed

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 $h(z) = 0$, we have:

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

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

This leads to:

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

- By remaining feasible always, *h(z) = 0*, $a \le z \le b$, one can apply an unconstrained algorithm (quasi-Newton) using (df/dz_S)

- Solve problem in reduced space of z_S variables

Example of Reduced Gradient

$$Min \quad x_1^2 - 2x_2$$

$$s.t. \quad 3x_1 + 4x_2 = 24$$

$$\nabla h^T = [3 \;\; 4], \;\; \nabla f^T = [2x_1 \;\; \text{-}\,2]$$

$$\text{Let } z_S = x_1, z_B = x_2$$

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

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

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

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

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 $h(z_S^k + \alpha d, z_B, z_N) = 0$
   - 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.

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 $h(z_S, z_B, z_N) = 0$ can get <u>very</u> expensive because ∇h is required
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 (MINOS/Augmented)

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

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

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

$$D^k z = c^k$$

3. Let $\psi = f(z) + v^T h(z) + \beta (h^T h)$ (Augmented Lagrange),
4. Solve linearly constrained problem:

$$\begin{aligned} &Min \qquad \psi(z) \\ &s.t. \qquad Dz = c \\ &\qquad\qquad a \le z \le b \end{aligned}$$

using reduced gradients to get z^{k+1}

5. Set $k=k+1$, go to 3.
6. Algorithm terminates when no movement between steps 3) 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

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

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.

<u>Notes</u>:
- 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.

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

<u>BFGS Formula</u> $\quad s = x^{k+1} - x^k$

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

- 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

$$\psi(x) = f(x) + uTg(x) + vTh(x)$$

$$+ \eta/2 \{\Sigma (h_j(x))^2 + \Sigma \max(0, g_j(x))^2\}$$

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 $\alpha \leq 1$

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:

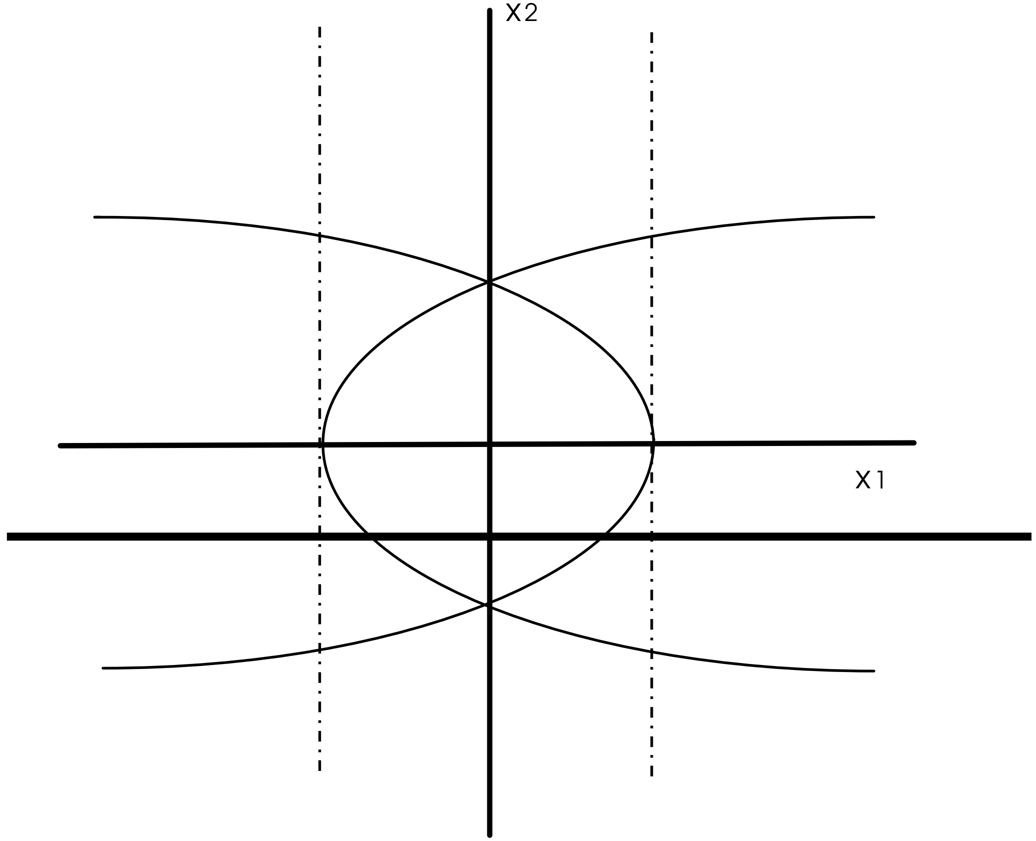
$$
\begin{aligned}
&\mathit{Min}_d \quad \nabla f(x^k)^T d + 1/2\, d^T B^k d \\
s.t. \quad &g(x^k) + \nabla g(x^k)^T d \le 0 \\
&h(x^k) + \nabla h(x^k)^T d = 0
\end{aligned}
$$

   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.

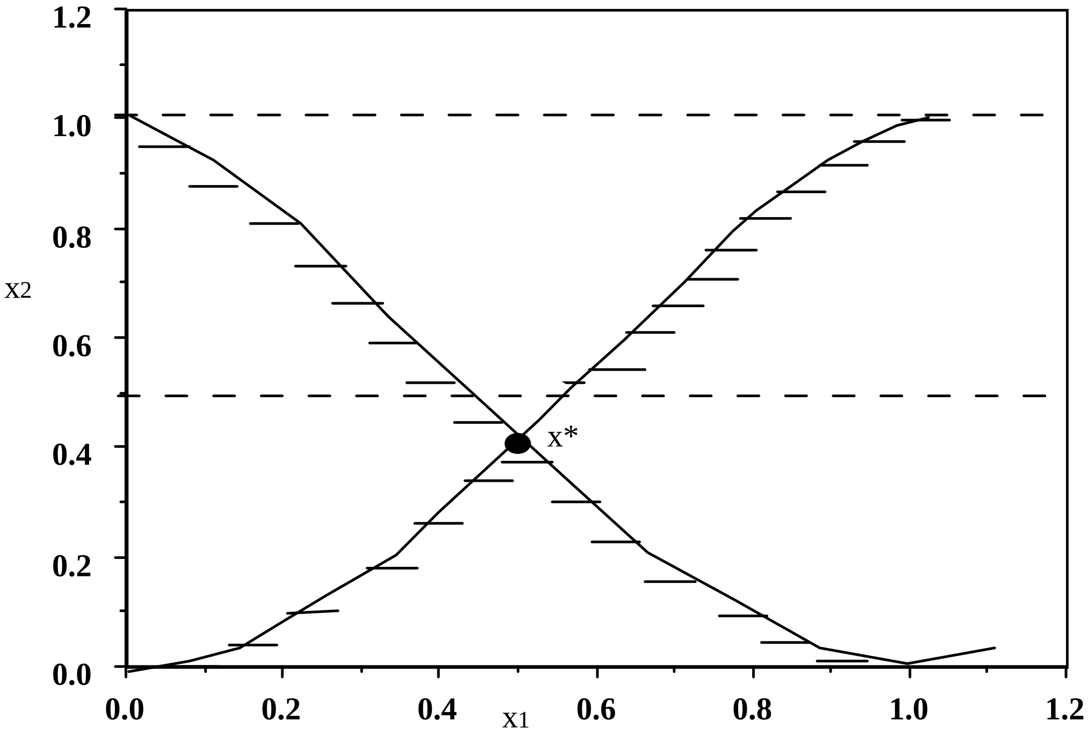
Problems with SQP

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



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

SQP Test Problem



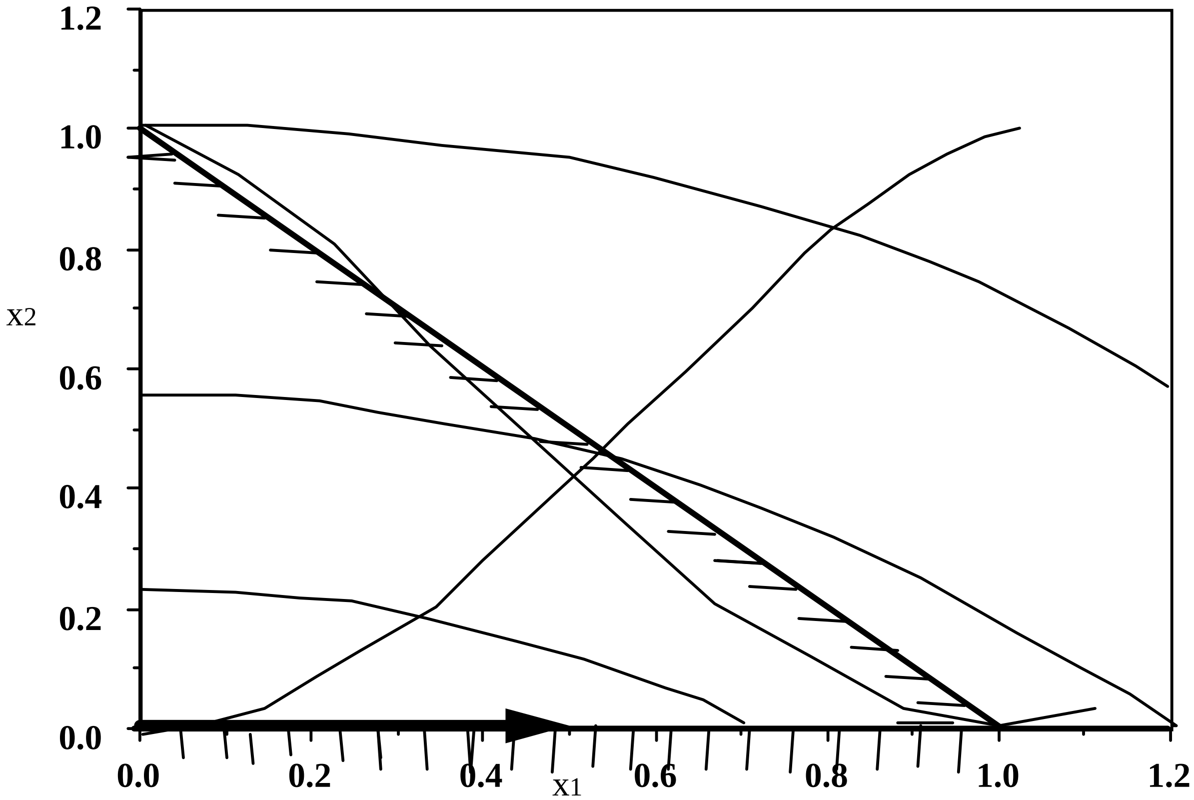
Min x2

s.t. $-x_2 + 2\,x_1^2 - x_1^3 \le 0$

$-x_2 + 2\,(1-x_1)^2 - (1-x_1)^3 \le 0$

$x^* = [0.5, 0.375]$.

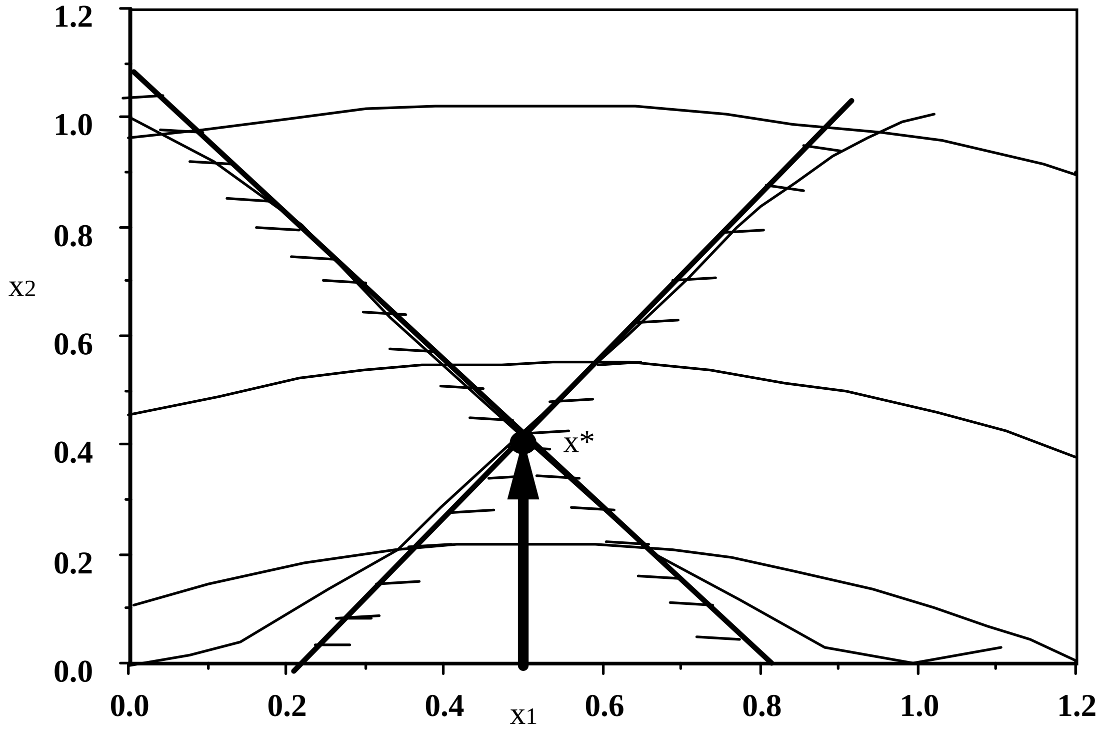
SQP Test Problem – First Iteration



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

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

SQP Test Problem – Second Iteration



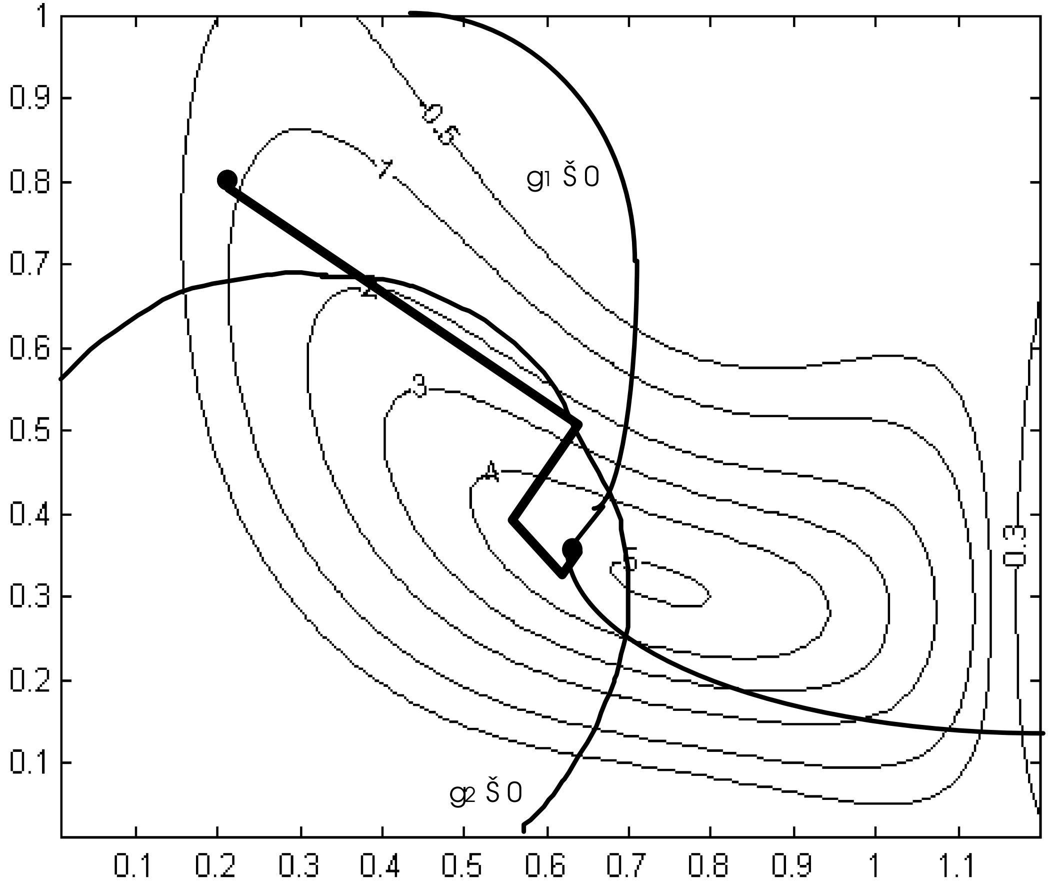
From $x_1 = [0.5, 0]^T$ with $B^1 = I$
(no update from BFGS possible), form:

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

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

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

Representative Constrained Problem SQP Convergence Path



<u>Starting Point</u>[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

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

$\Rightarrow$As μ ➔ 0, $x^*(\mu)$ ➔ x^* Fiacco and McCormick (1968)

Solution of the Barrier Problem

⇒Newton Directions (KKT System)

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

⇒ 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.

$$\text{for } \alpha \in (0, \bar{\alpha}],\; x_{k+1} = x_k + \alpha \, d_x$$

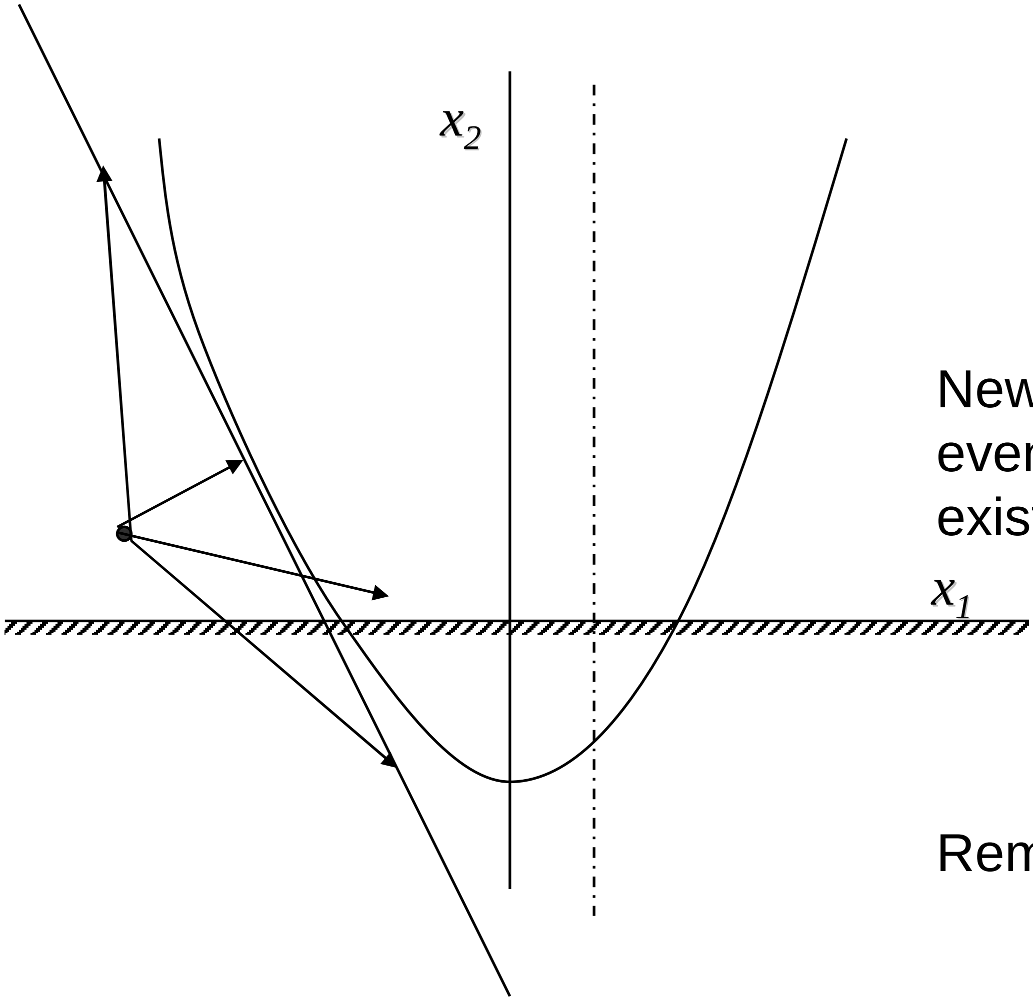
$$x_k + \bar{\alpha} \, d_x \geq (1-\tau) x_k > 0$$

$$v_{k+1} = v_k + \bar{\alpha} \, d_v \geq (1-\tau) v_k > 0$$

$$\lambda_{k+1} = \lambda_k + \alpha \, (\lambda_+ - \lambda_k)$$

- 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

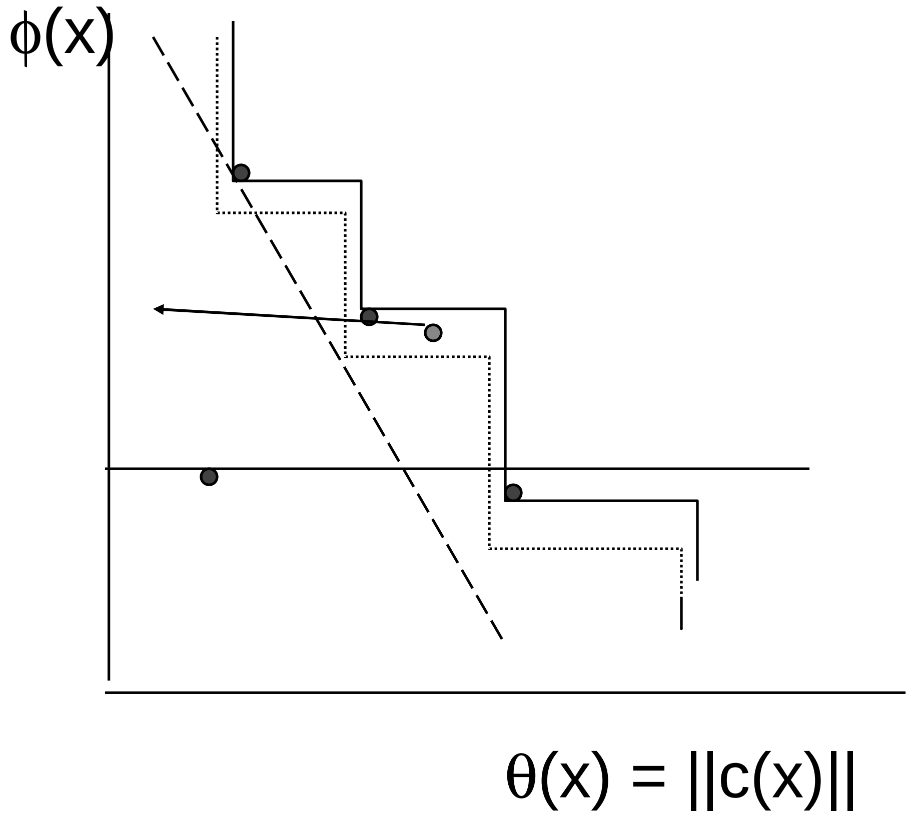
If switching condition

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

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 nonsingular

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

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

KKT matrix factored by MA27

Feasibility restoration phase

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

$$x_l \le x_k \le x_u$$

Apply Exact Penalty Formulation

Exploit same structure/algorithm to reduce infeasibility

Details of IPOPT Algorithm

Options

Line Search Strategies

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

Hessian Calculation

- BFGS (reduced space)
- SR1 (reduced space)
- Exact full Hessian (direct)
- Exact reduced Hessian (direct)
- Preconditioned CG

Comparison

34 COPS Problems

(600 - 160 000 variables)

486 CUTE Problems

(2 – 50 000 variables)

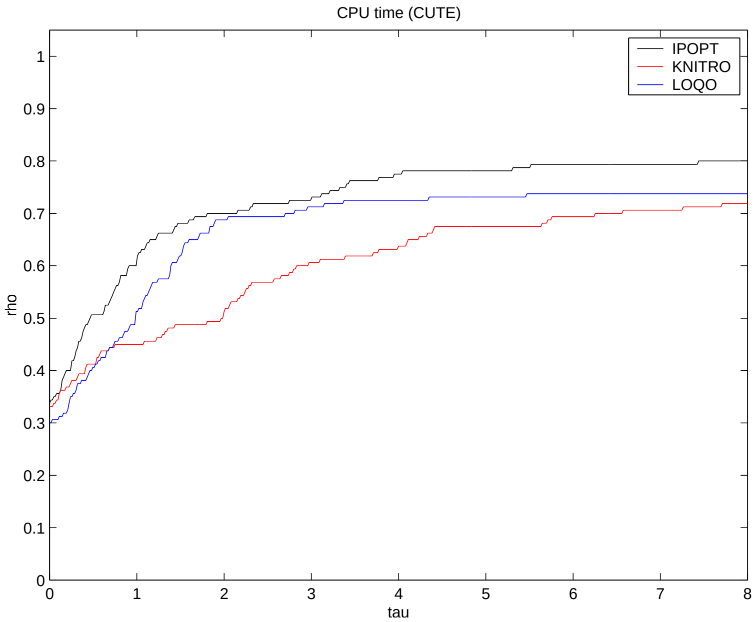
56 MITT Problems

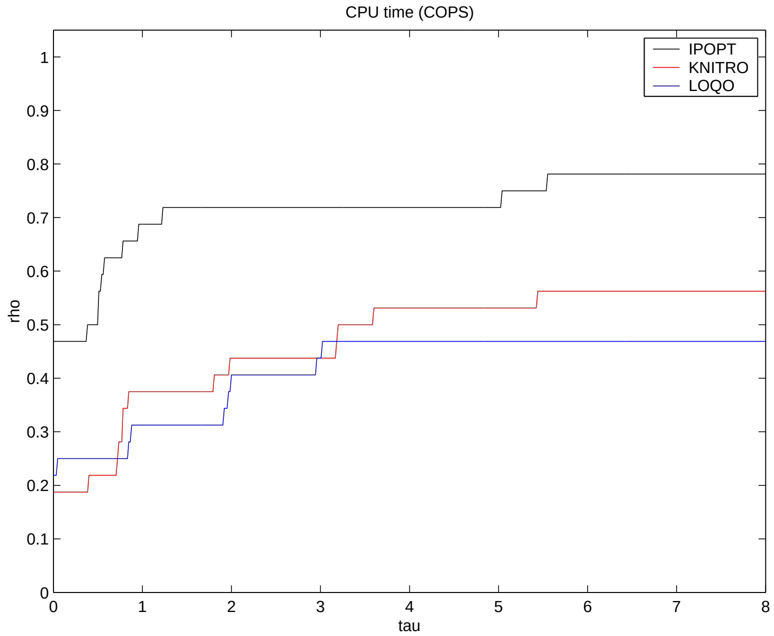
(12097 – 99998 variables)

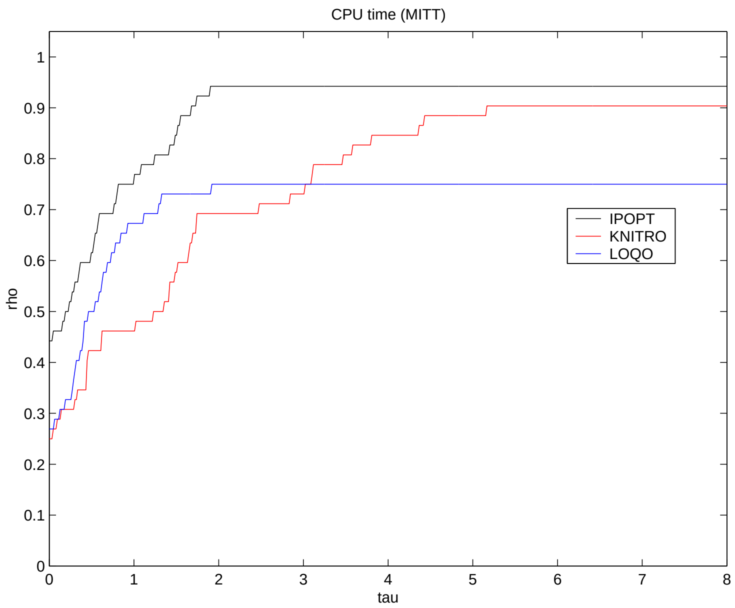
Performance Measure

- $r_{p,l}$ = (#iter$_{p,l}$)/ (#iter$_{p,min}$)
- $P(\tau)$ = fraction of problems with $\log_2(r_{p,l}) < \tau$

IPOPT Comparison with KNITRO and LOQO



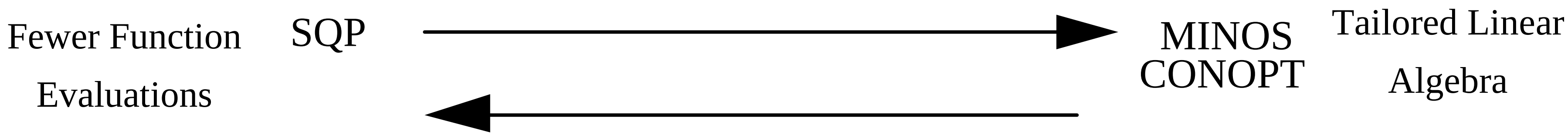




- IPOPT has better performance, robustness on CUTE, MITT and COPS problem sets
- Similar results appear with iteration counts
- Can be downloaded from http://www.coin-or.org
- See links for additional information

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



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

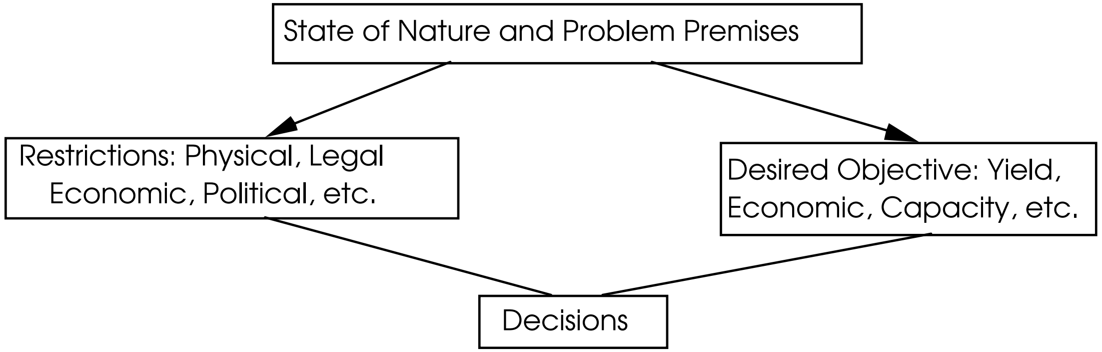
MS Excel

Solver uses Generalized Reduced Gradient method with restoration

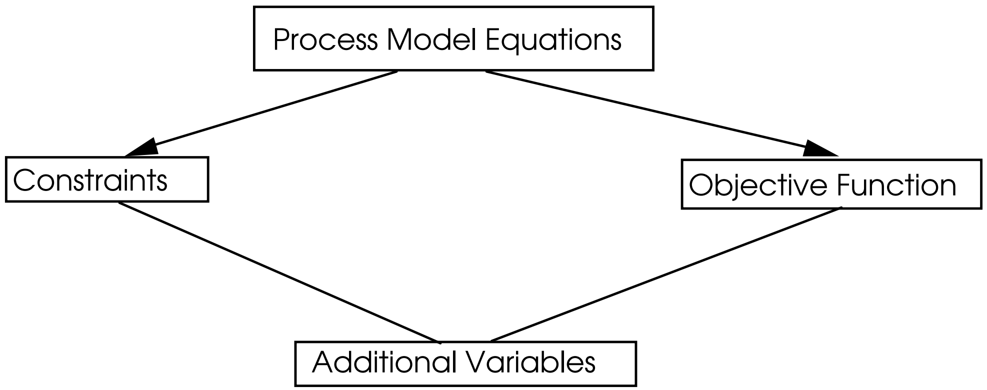
Rules for Formulating Nonlinear Programs

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. $\mathit{Min} \quad [Tx - 3Ty]$
   s.t. $xT + y - T^2 y = 5$
   $4x - 5Ty + Tx = 7$
   $0 \leq T \leq 1$
   (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



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++

1999- : Simultaneous dynamic optimization
over 1 000 000 variables and constraints

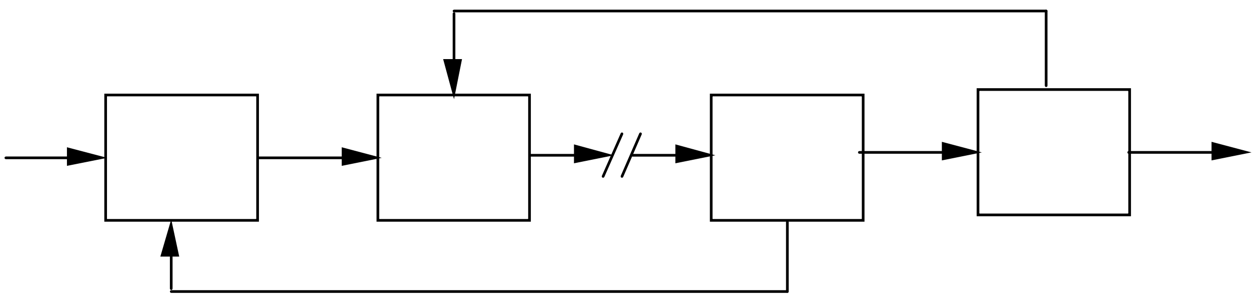
Current: Tailor structure, architecture and problems

Flowsheet Optimization Problems - Introduction

Modular Simulation Mode

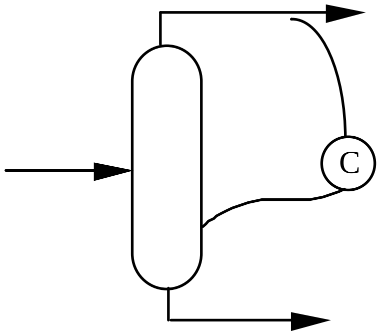
Physical Relation to Process

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



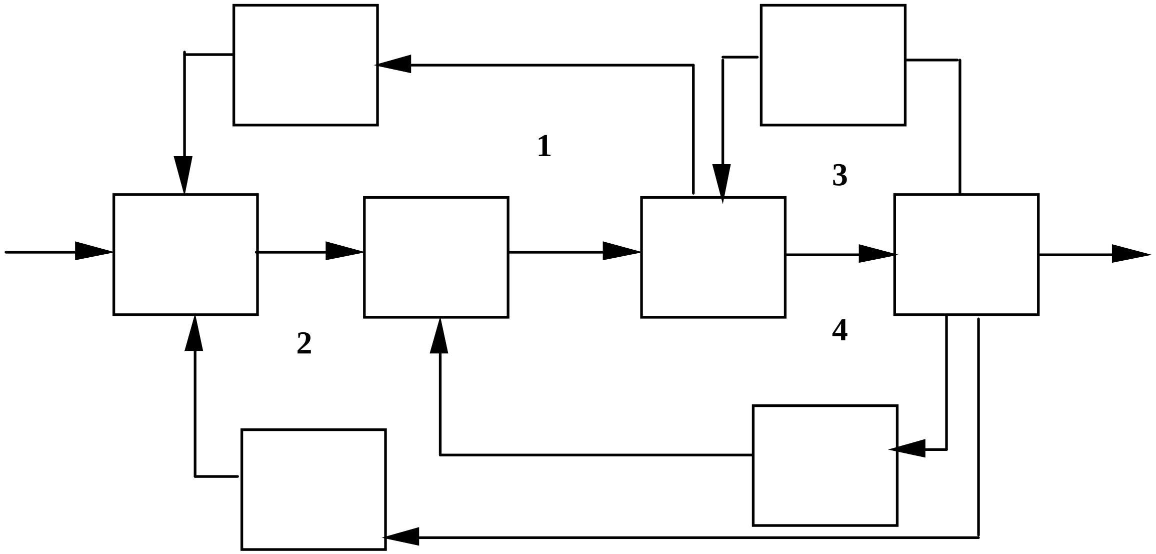
•Convergence Procedures - for simple flowsheets, often identified from flowsheet structure
•Convergence "mimics" startup.
•Debugging flowsheets on "physical" grounds

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 looks

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

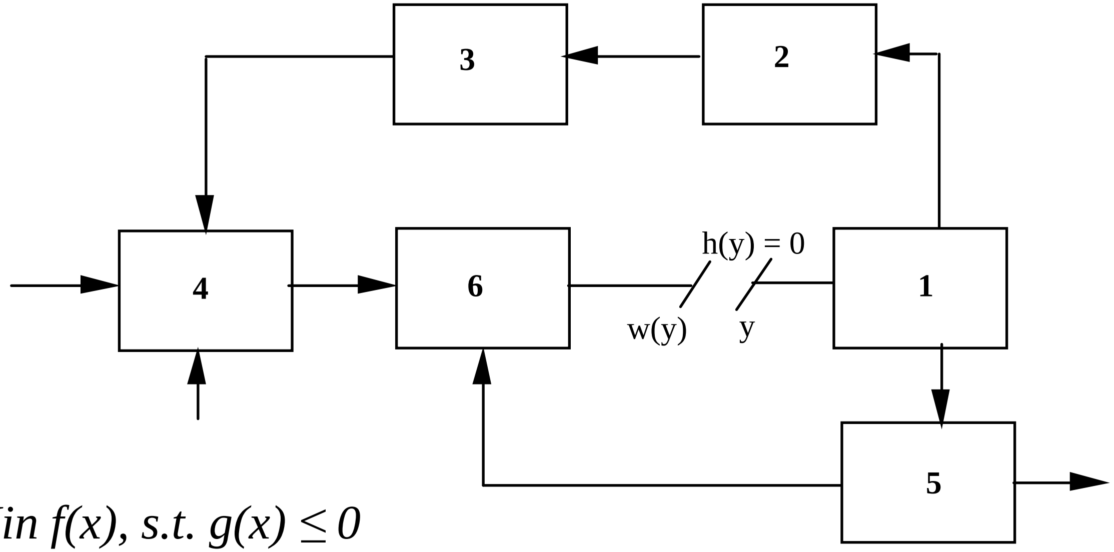
Pro/II

ProSim

ROMeo

VTPLAN

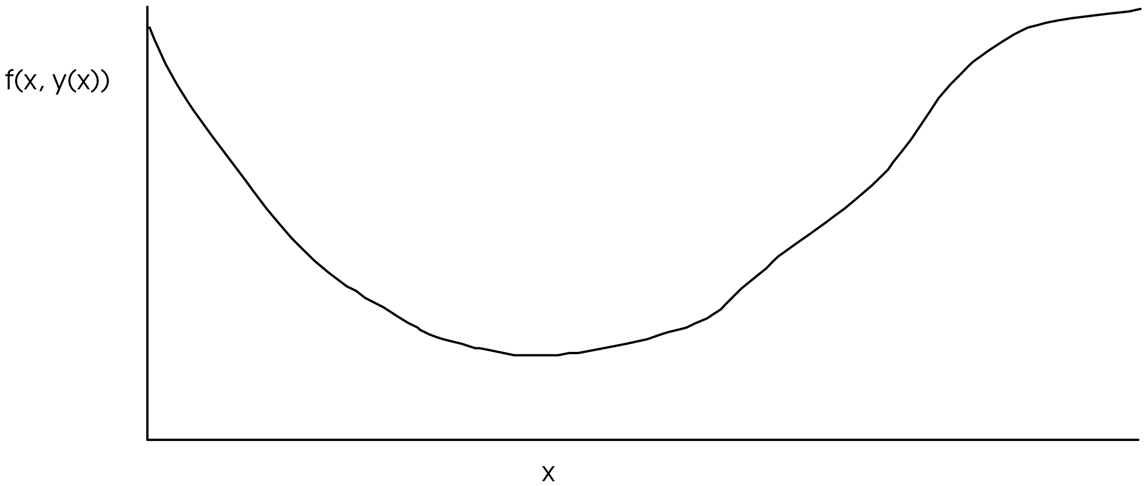
Simulation and Optimization of Flowsheets



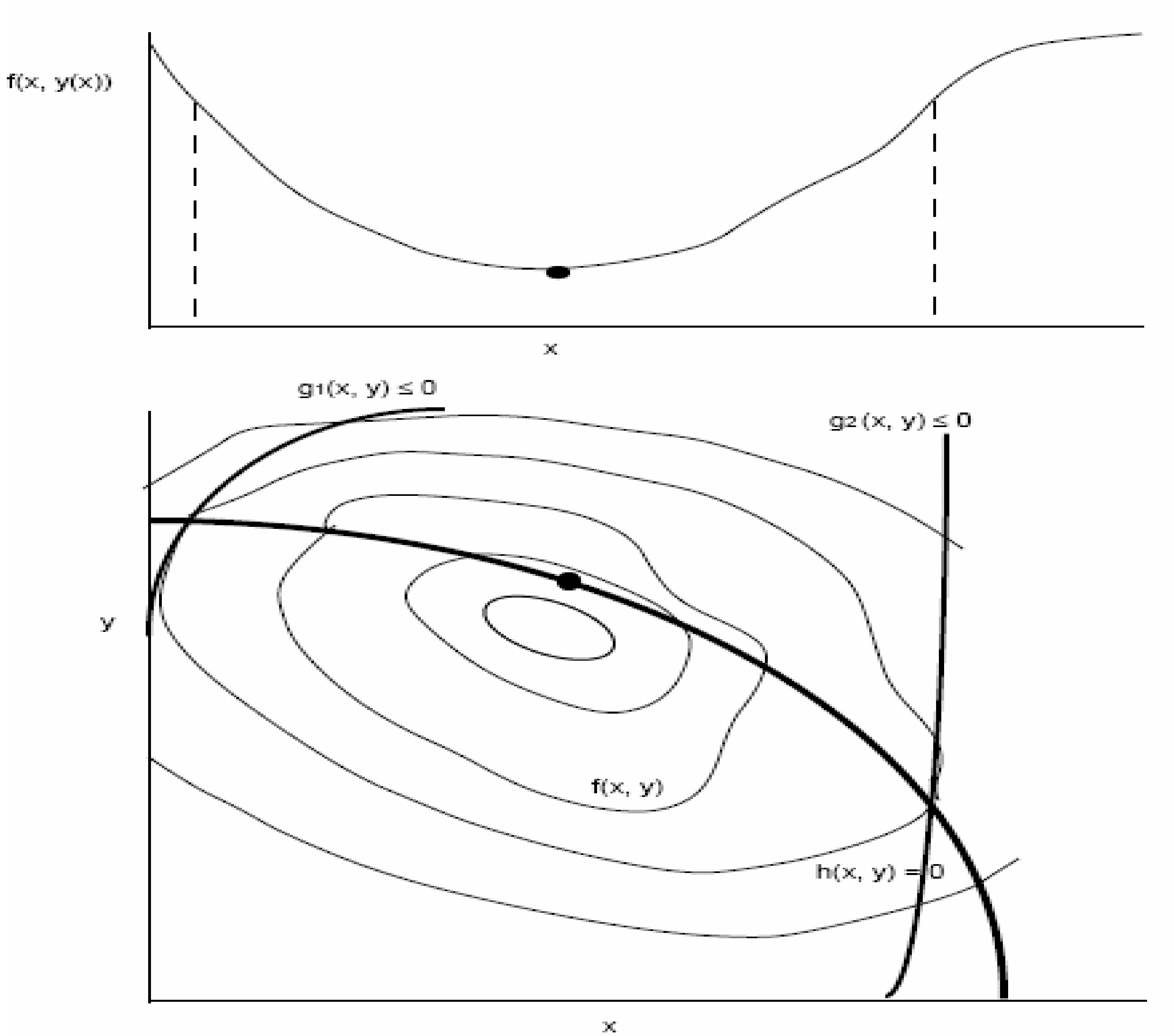
Min f(x), s.t. g(x) ≤ 0

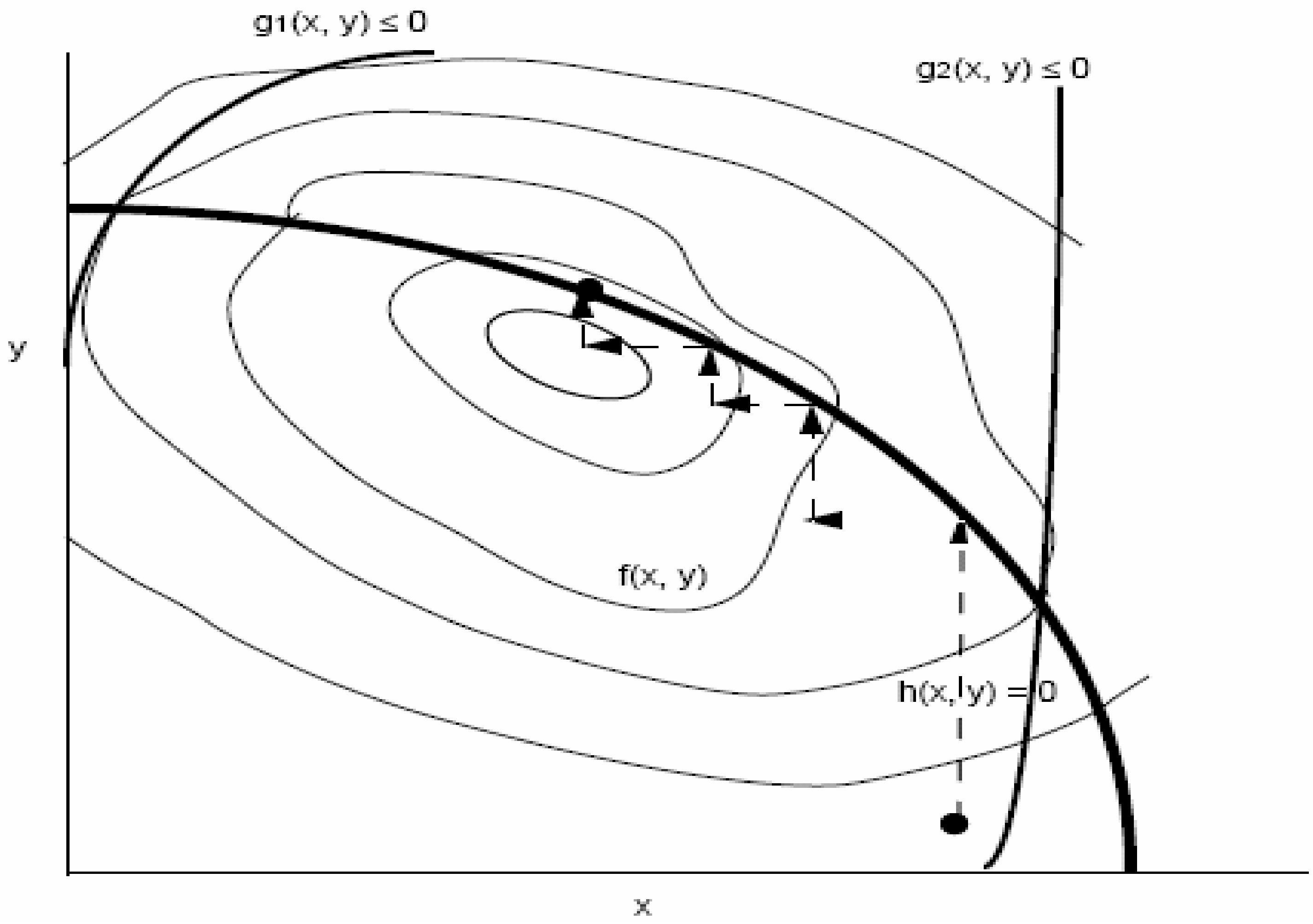
For single degree of freedom:

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



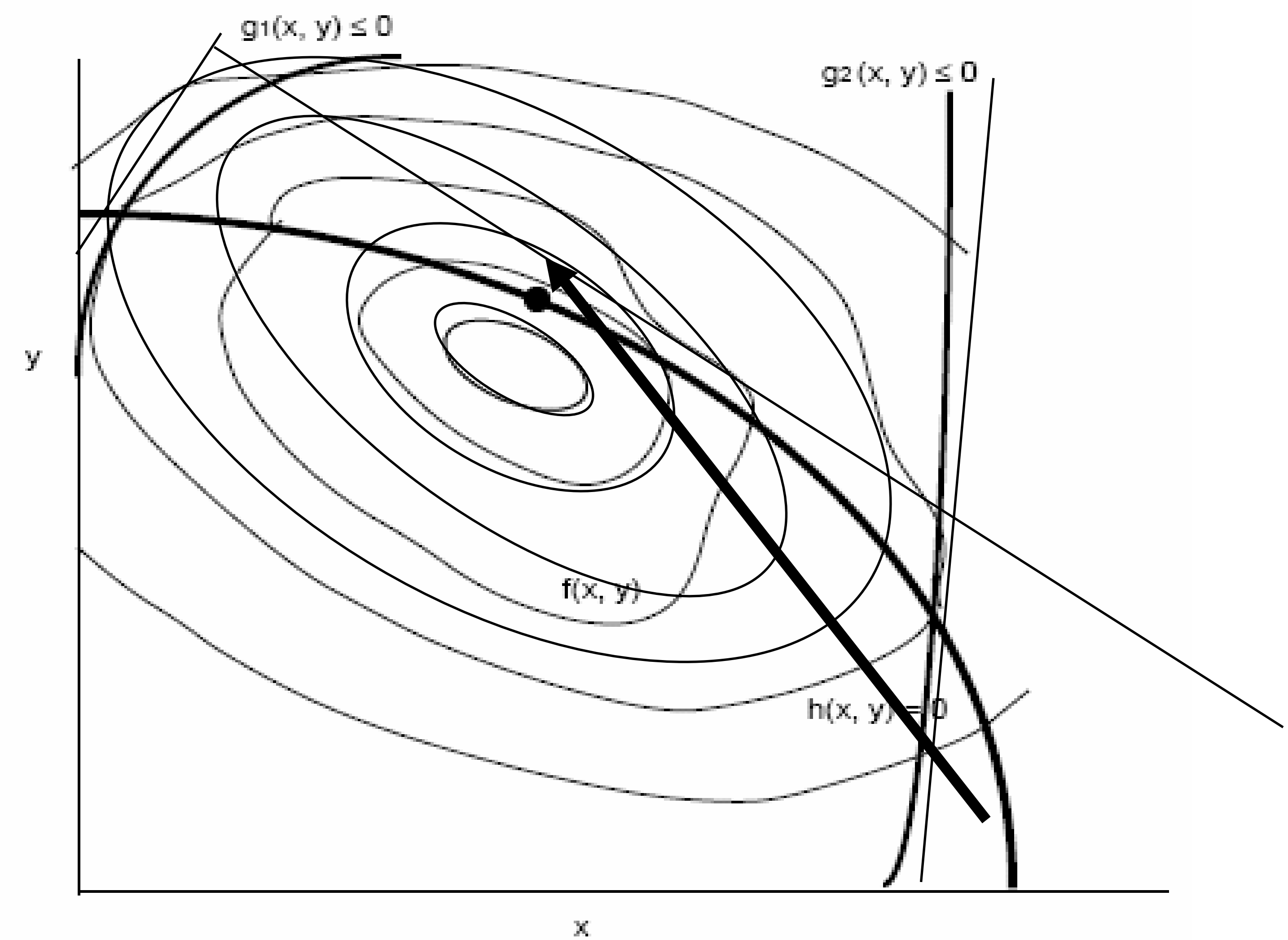
Expanded Region with Feasible Path





"Black Box" Optimization Approach

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



<u>SQP - Infeasible Path Approach</u>

- solve and optimize simultaneously in x <u>and</u> 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.

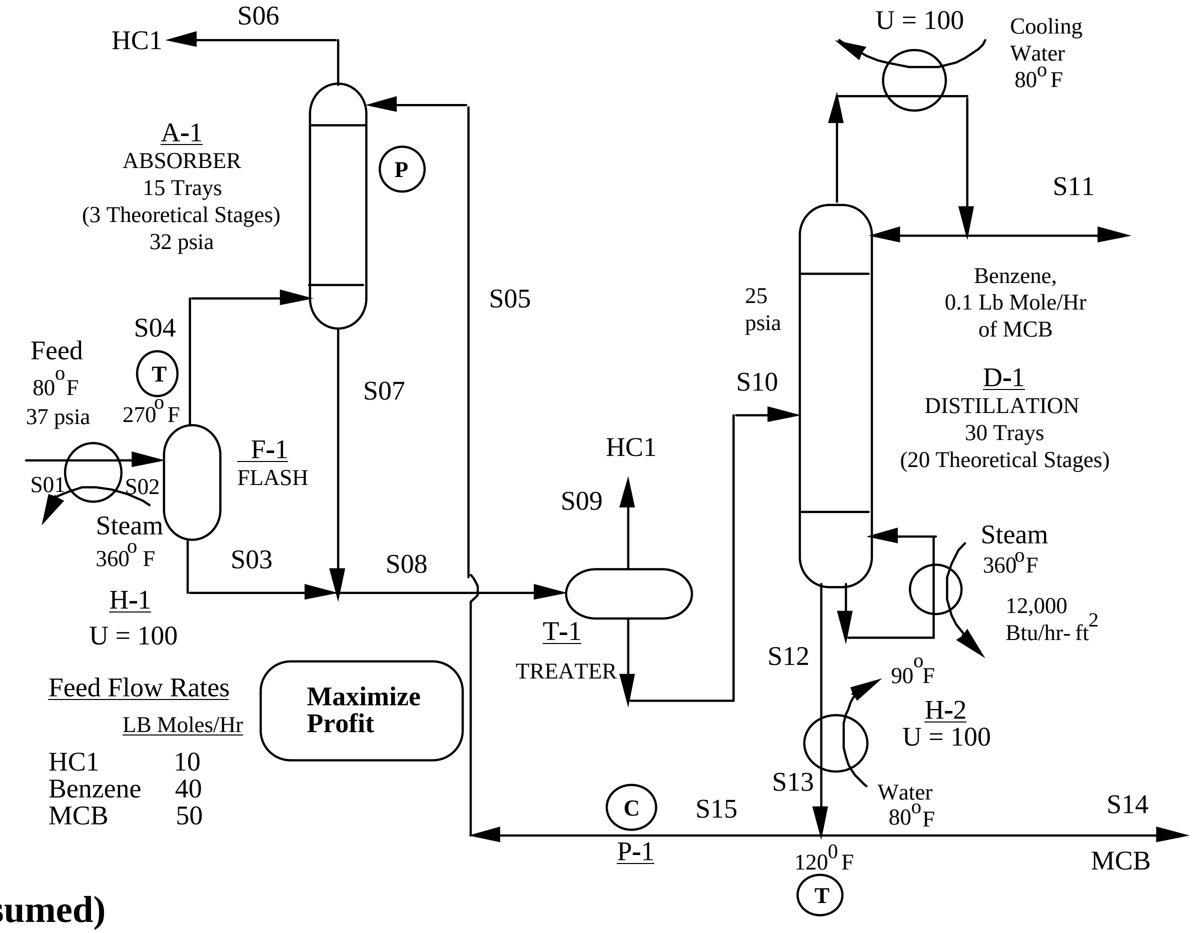
Examples

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

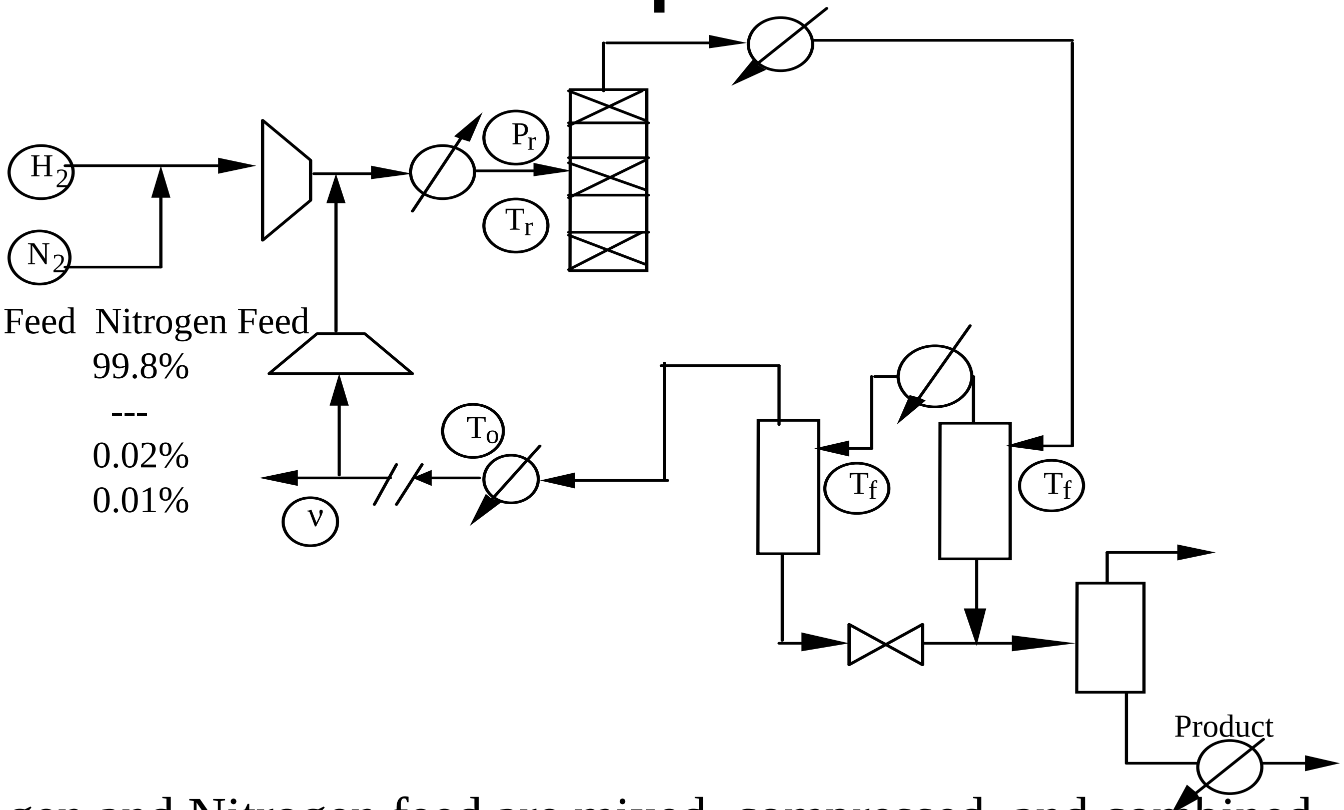
Optimization of Monochlorobenzene Process



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. • 105 tons NH3/yr.
- Pressure Balance
- No Liquid in Compressors
- $1.8 \leq H2/N2 \leq 3.5$
- Treact $\leq$ 1000o F
- NH3 purged $\leq$ 4.5 lb mol/hr
- NH3 Product Purity $\geq$ 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

	Optimum	Starting point
Objective Function($\$10^6$)	24.9286	20.659
1. Inlet temp. reactor (^oF)	400	400
2. Inlet temp. 1st flash (^oF)	65	65
3. Inlet temp. 2nd flash (^oF)	35	35
4. Inlet temp. rec. comp. (^oF)	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?

<u>Recognizing True Solution</u>

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

<u>Performance of Algorithms</u>

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

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

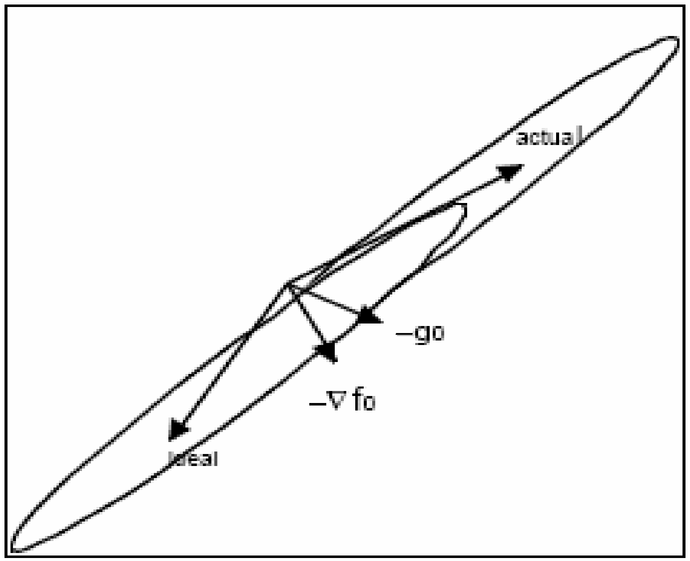
$$Min\ f(x) = x^T A x$$

$$A = \begin{bmatrix} \varepsilon + 1/\varepsilon & \varepsilon - 1/\varepsilon \\ \varepsilon - 1/\varepsilon & \varepsilon + 1/\varepsilon \end{bmatrix}$$

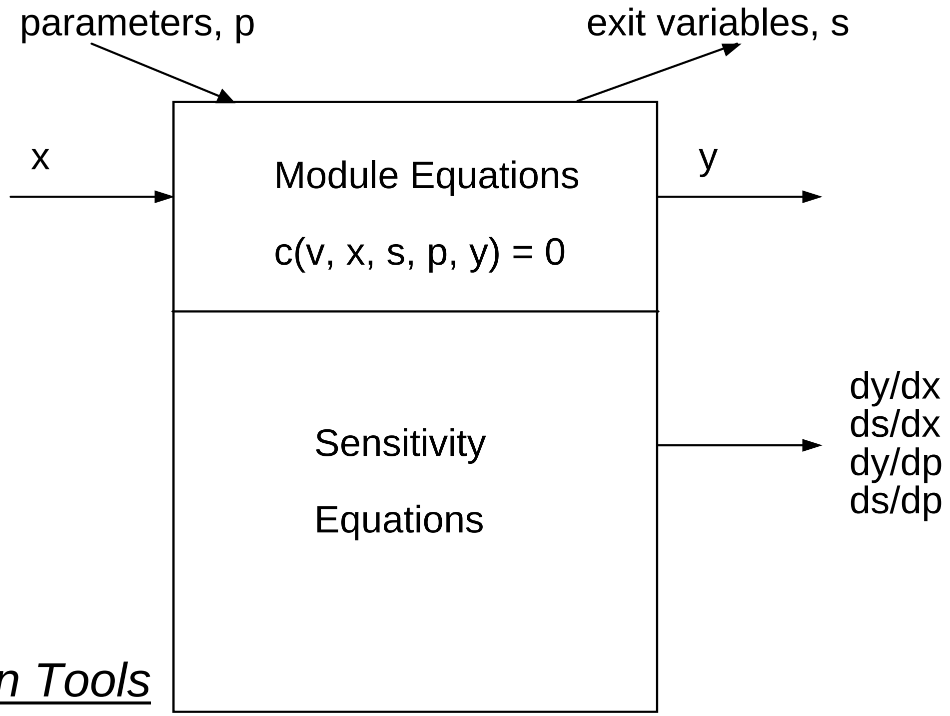
$$x_0 = [1 \quad 1]^T \quad \nabla f(x_0) = \varepsilon\, x_0$$

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

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



Implementation of Analytic Derivatives



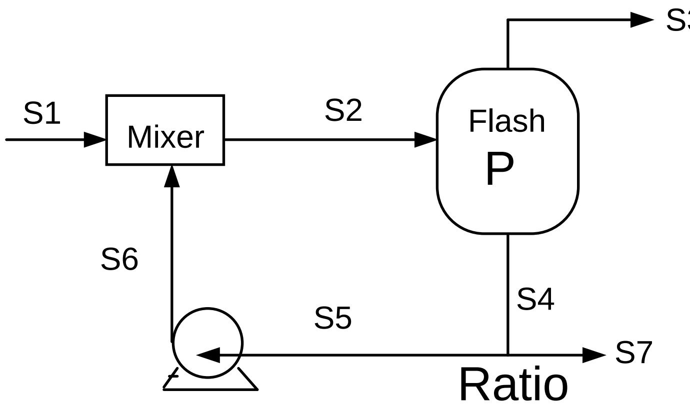
Automatic Differentiation Tools

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

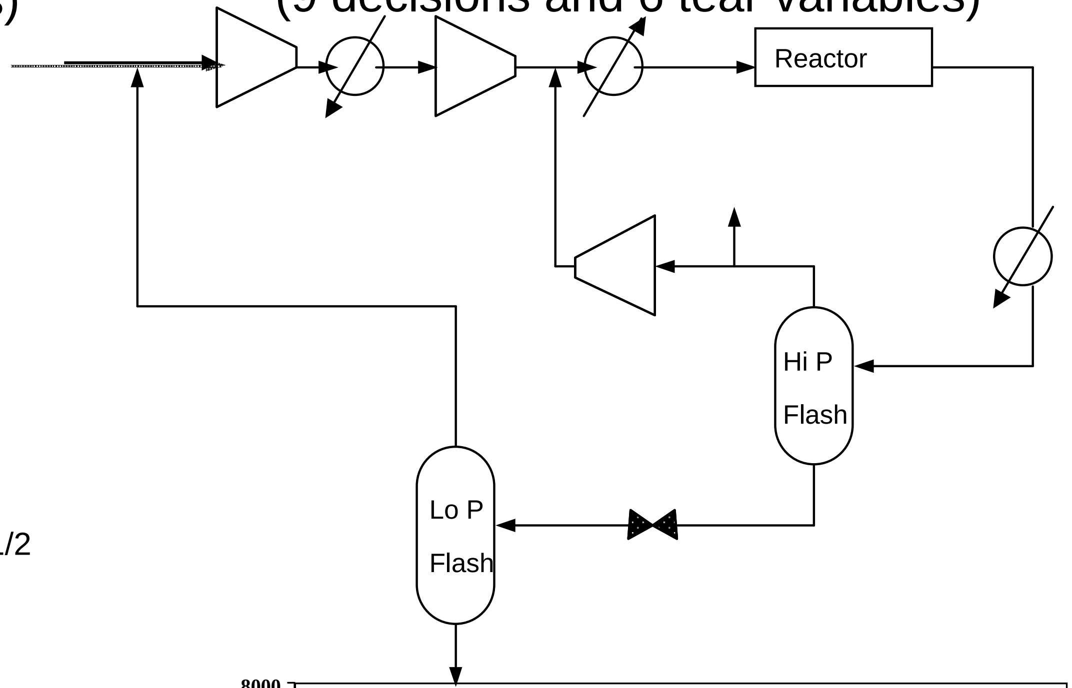
Relative effort needed to calculate gradients is not n+1 but about 3 to 5 (Wolfe, Griewank)

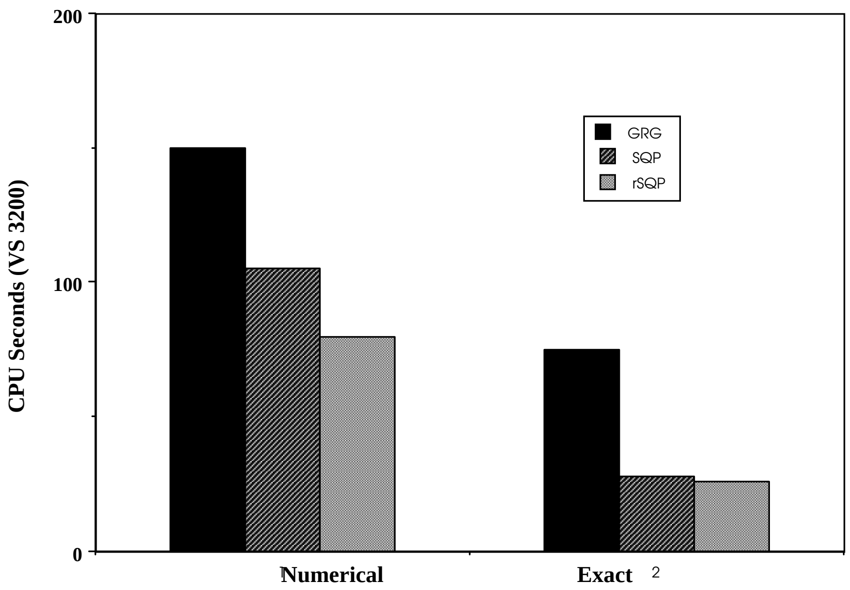
Flash Recycle Optimization
(2 decisions + 7 tear variables)

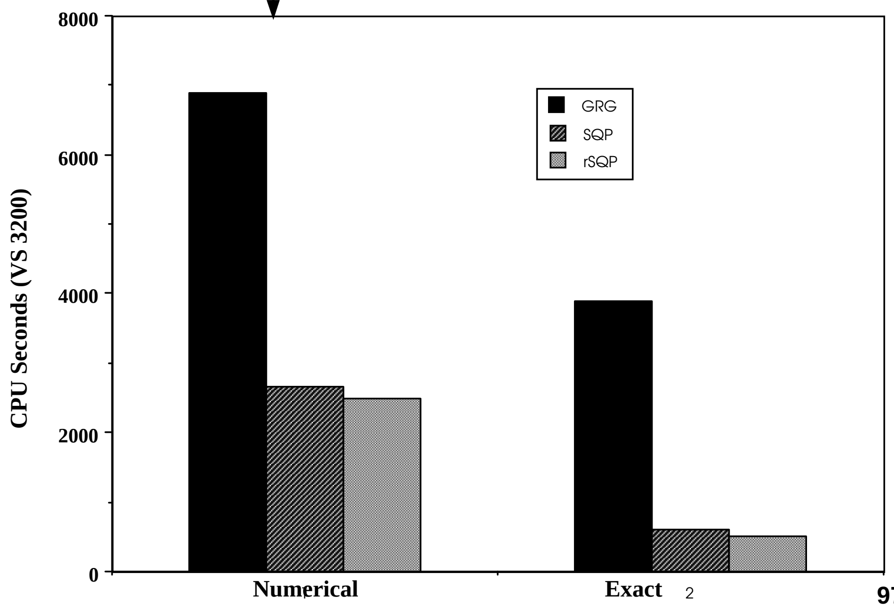
Ammonia Process Optimization
(9 decisions and 6 tear variables)



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







Large-Scale SQP

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

<u>Characteristics</u>

- Many equations and variables ($\geq$ 100 000)
- Many bounds and inequalities ($\geq$ 100 000)

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

<u>Many degrees of freedom ($\geq$ 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:

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

- 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:

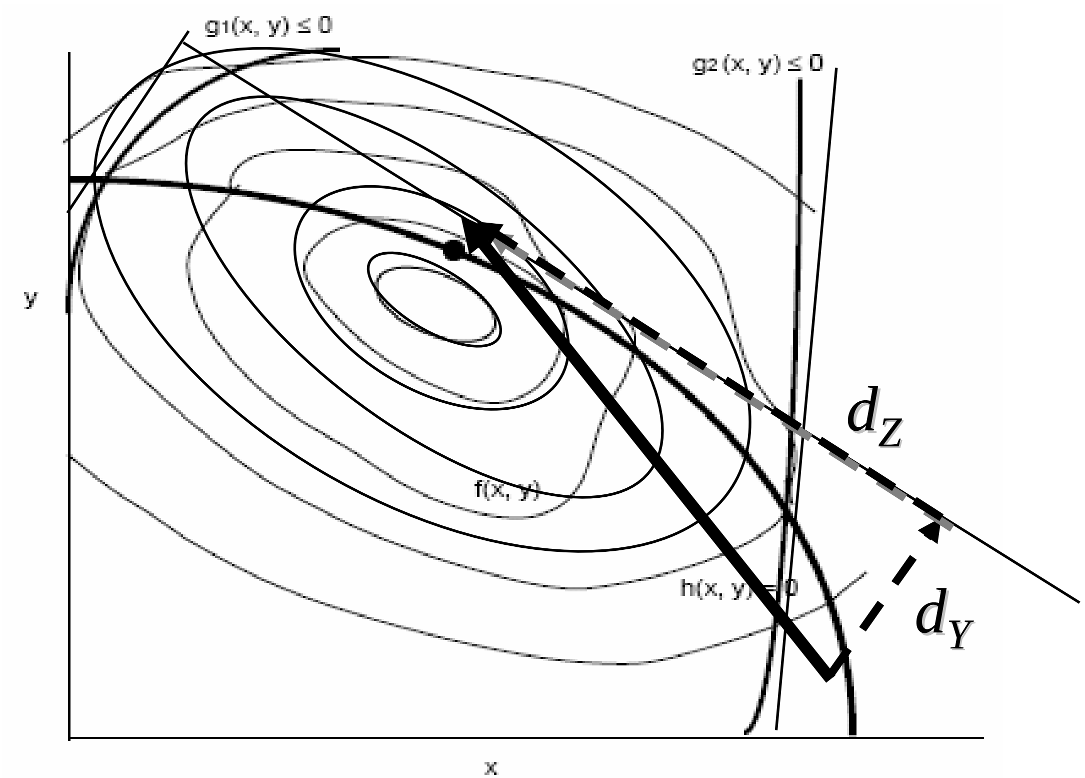
$$\begin{bmatrix} [Y^k\ Z^k]^T & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} W^k & A^k \\ A^{k^T} & 0 \end{bmatrix} \begin{bmatrix} [Y^k\ Z^k] & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} d_Y \\ d_Z \\ \lambda_+ \end{bmatrix} = -\begin{bmatrix} [Y^k\ Z^k]^T & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \nabla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

Reduced space SQP (rSQP)
Range and Null Space Decomposition

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

- ***($A^T Y$) 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 --> 0)
- ***($Y^T A$) λ = -$Y^T \nabla f(x^k)$***, λ can be determined by first order estimate
- Calculate or approximate $w = Z^T W Y\, d_Y$, solve ***$Z^T W Z\, d_Z = -Z^T \nabla f(x^k) - w$***
- Compute total step: ***$d = Y\, dY + Z\, dZ$***

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
   $(A^T Y)\, d_Y = -c(x^k)$
5. Update projected Hessian $B^k \sim Z^T W Z$ (e.g. with BFGS), w_k (e.g., zero)
6. Solve small QP for step d_Z in Null space.

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

7. If error is less than tolerance stop. Else
8. Solve for multipliers using $(Y^T A)\, \lambda = -Y^T \nabla f(x^k)$
9. Calculate total step $d = Y\, d_Y + Z\, d_Z$.
10. Find step size α and calculate new point, $x_{k+1} = x_k +$
11. 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	50	42	42				
(a)				4.4	144	3.2 (2.1/1.1)	23
(b)				4.7	157	2.8 (1.6/1.2)	13
Distill'n Ideal	228	227	227				
(a)				28.5	24	38.6 (9.6/29.0)	7
(b)				33.5	58	69.8 (17.2/52.6)	14
Distill'n Nonideal	569	567	567				
(1)				172.1	34	130.1 (47.6/82.5)	14
(a)				432.1	362	144.9 (132.6/12.3)	47
(b)				855.3	745	211.5 (147.3/64.2)	49
(c)							
Distill'n Nonideal	977	975	975				
(2)				(F)	(F)	230.6 (83.1/147.5)	9
(a)				520.0+	162	322.1 (296.4/25.7)	26
(b)				(F)	(F)	466.7 (323/143.7)	34
(c)							

* CPU Seconds - VAX 6320

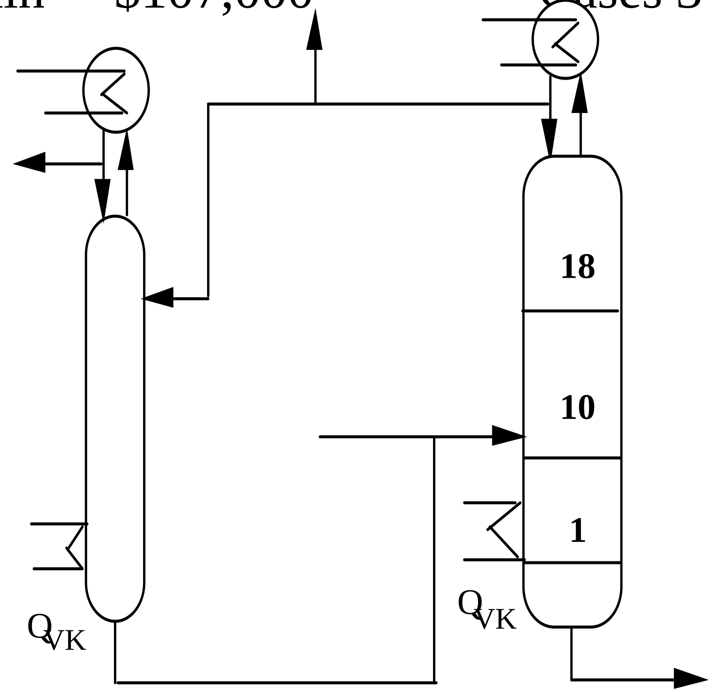
\+ MINOS (5.1)

(F) Failed

Comparison of SQP and rSQP

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

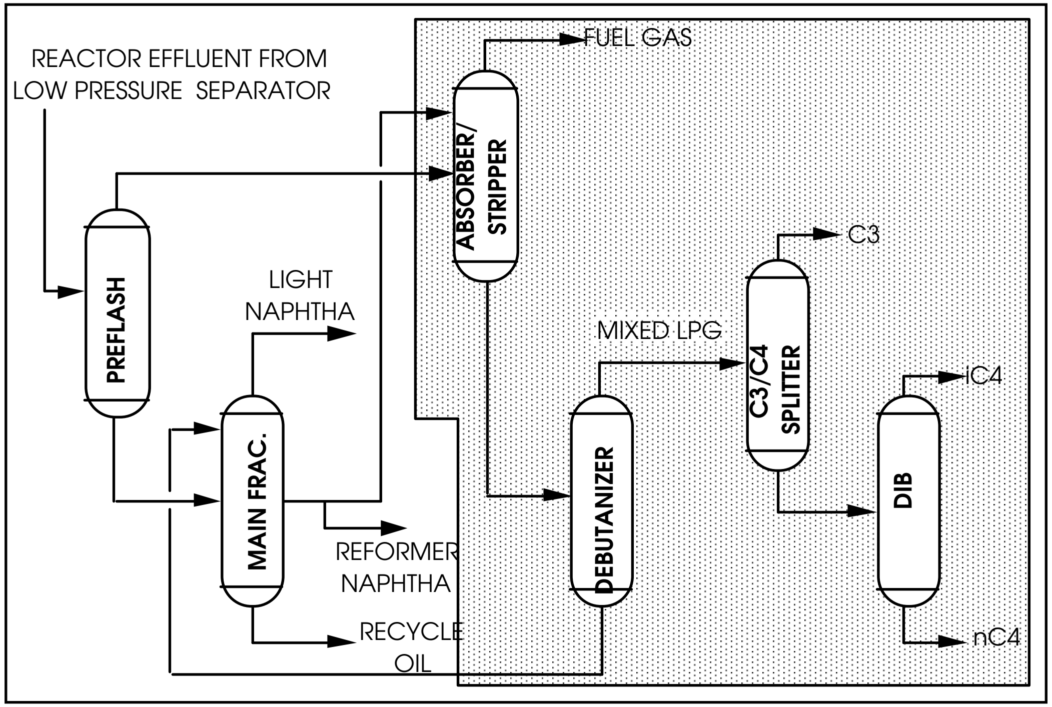
	Method	CPU Time	Annual Savings	Comments
1.	SQP*	2 hr	negligible	Base Case
2.	rSQP	15 min.	$ 42,000	Base Case
3.	rSQP	15 min.	$ 84,000	Higher Feed Tray Location
4.	rSQP	15 min.	$ 84,000	Column 2 Overhead to Storage
5.	rSQP	15 min	$107,000	Cases 3 and 4 together



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
$\dfrac{\textbf{Time rSQP}}{\textbf{Time MINOS}}$ **(%)**	12.8%	12.7%	10.7%	7.3%	5.7%	16.1%

Many degrees of freedom => full space IPOPT

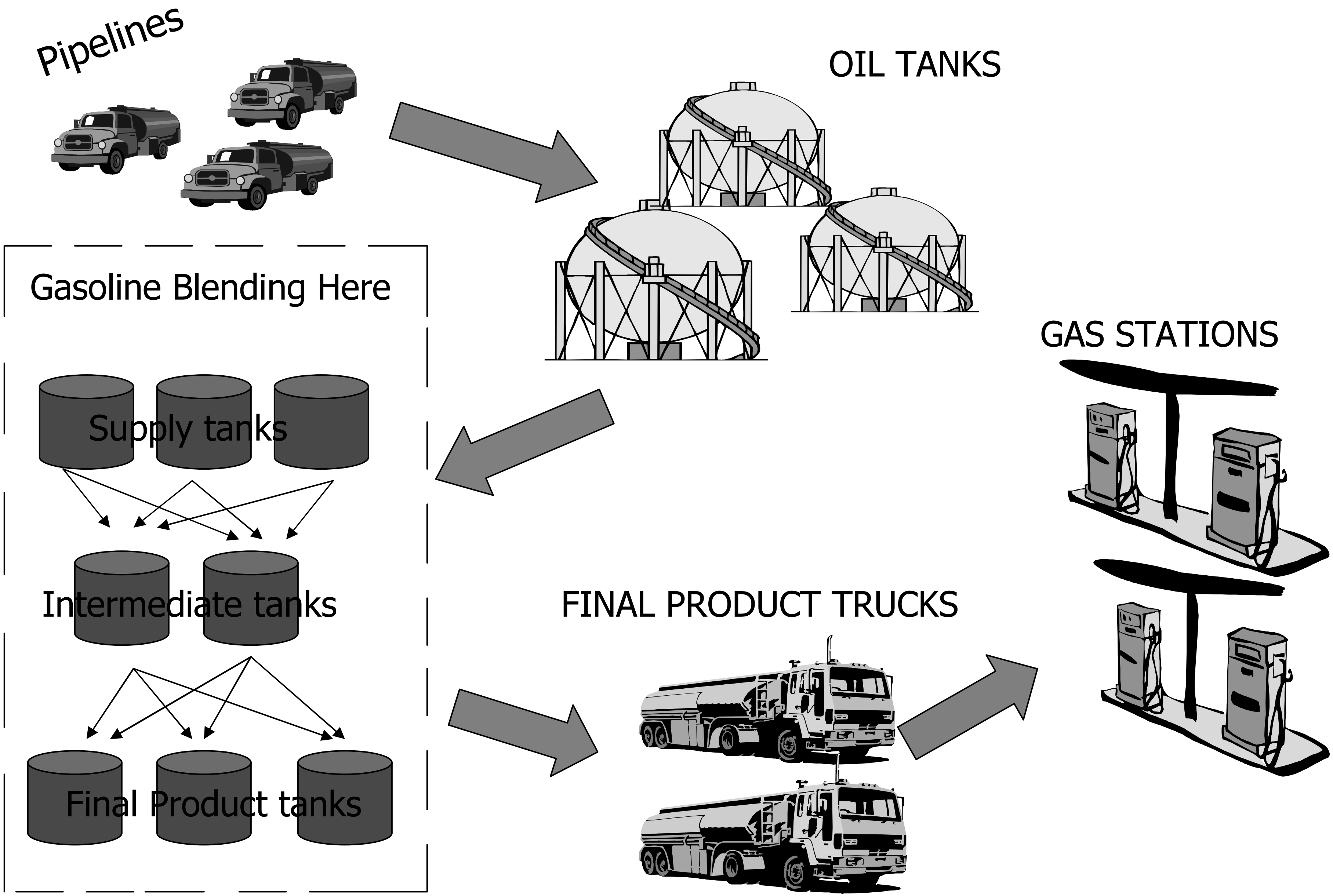
$$\begin{bmatrix} W^k + \Sigma & A^k \\ A^{kT} & 0 \end{bmatrix} \begin{bmatrix} d \\ \lambda_+ \end{bmatrix} = -\begin{bmatrix} \nabla \varphi(x^k) \\ c(x^k) \end{bmatrix}$$

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

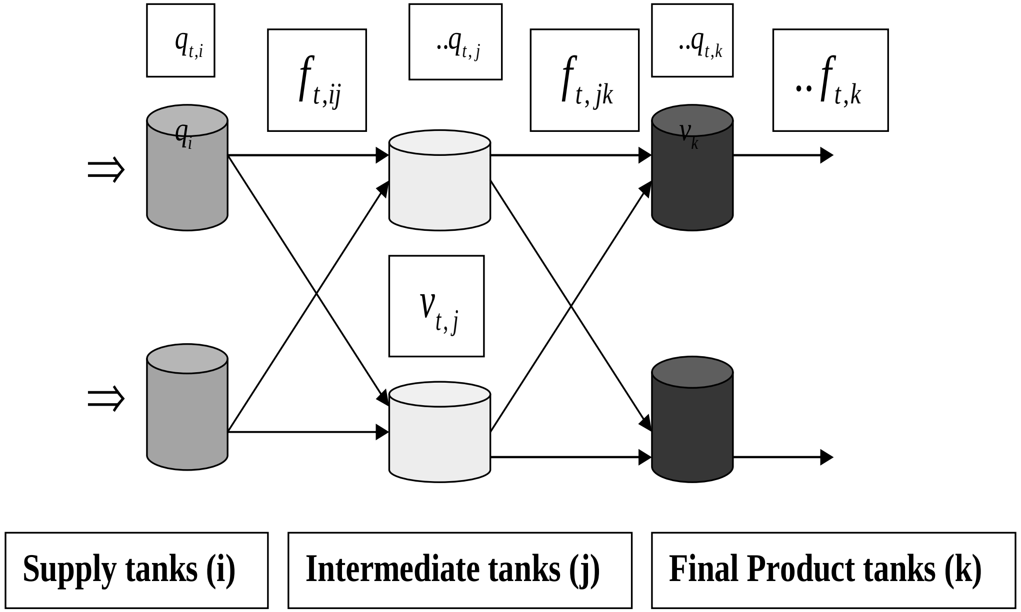
+ $W = \nabla_{xx} L(x,\lambda)$ and $A = \nabla c(x)$ sparse, often structured
+ fast if many degrees of freedom present
+ no variable partitioning required

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

Gasoline Blending



Blending Problem & Model Formulation



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

$$\text{s.t.} \quad \sum_{k} f_{t,jk} - \sum_{i} f_{t,ij} + v_{t+1,j} = v_{t,j}$$

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

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

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

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

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

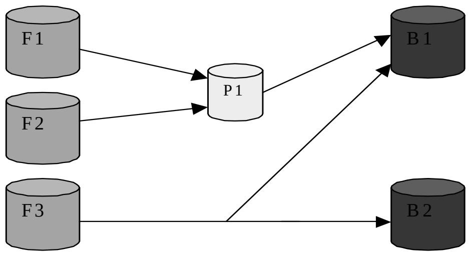
f & v ------ flowrates and tank volumes
q ------ tank qualities

Model Formulation in AMPL

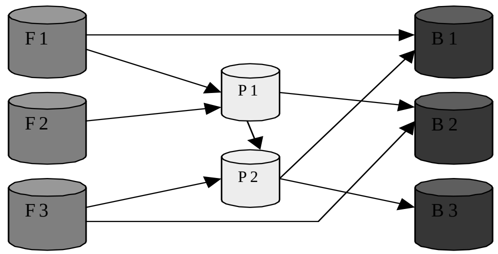
Small Multi-day Blending Models

Single Qualities

Haverly, C. 1978 (HM)



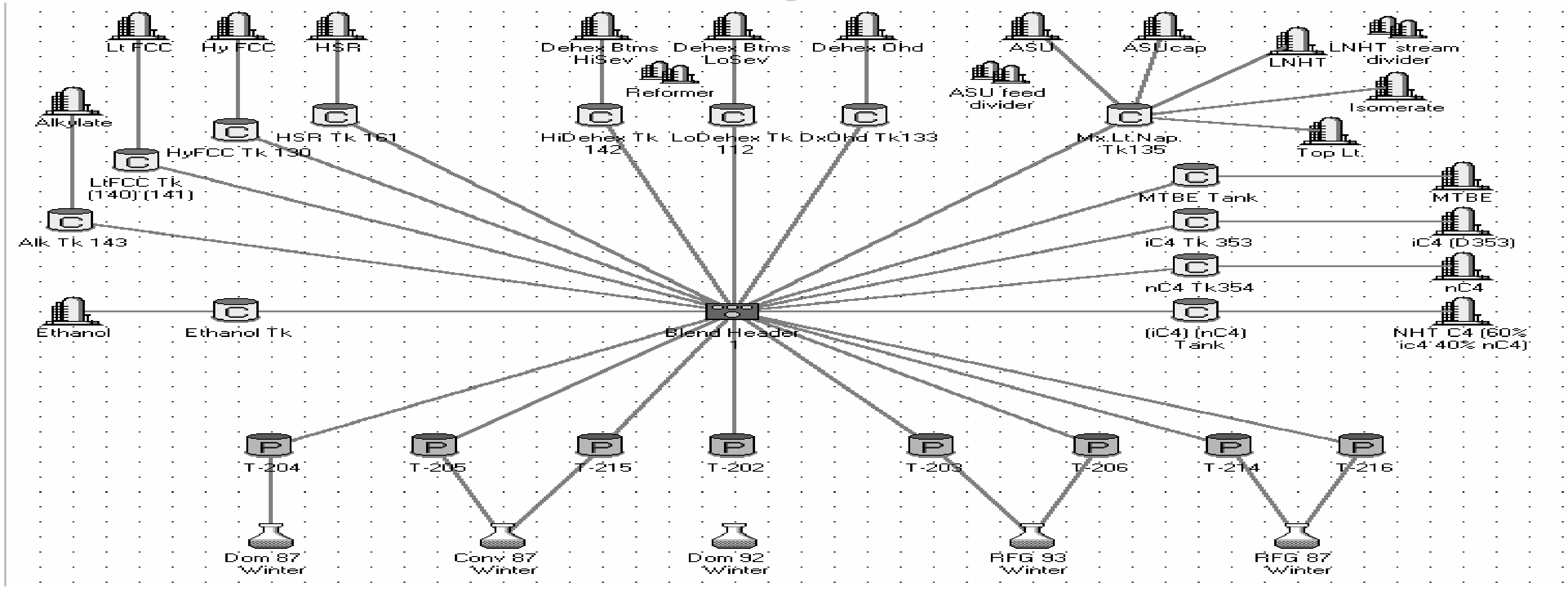
Audet & Hansen 1998 (AHM)



	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

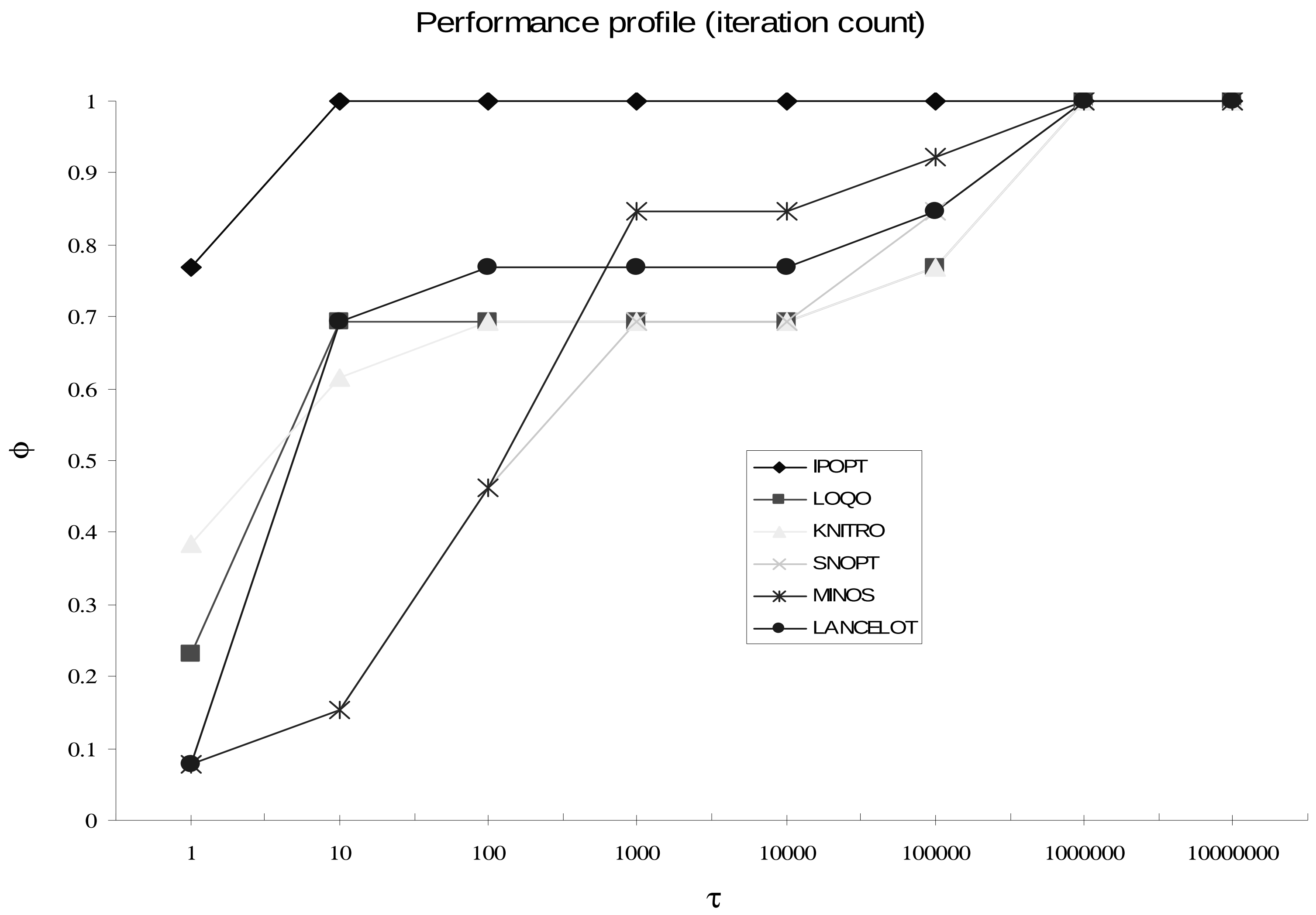
Honeywell Blending Model – Multiple Days
48 Qualities



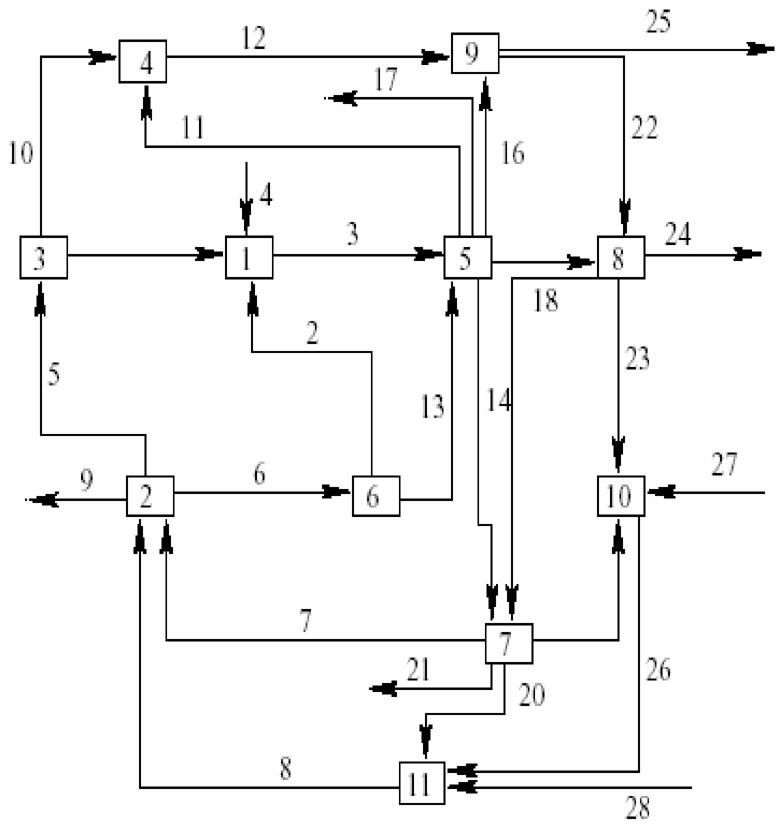
	no. of iterations	objective	CPU (s)	normalized CPU (s)
IHM Day 1 ($N = 2003, M = 1595, S = 1449$)				
LANCELOT	388	6.14×10^1	1.17×10^5	5.28×10^3
MINOS	2238	6.14×10^1	5.24×10^1	1.66×10^2
SNOPT	*a*	*a*	*a*	*a*
KNITRO	37	1.00×10^2	1.58×10^2	7.11×10^1
LOQO	*b*	*b*	*b*	*b*
IPOPT, exact	21	6.14×10^1	2.60	2.60
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

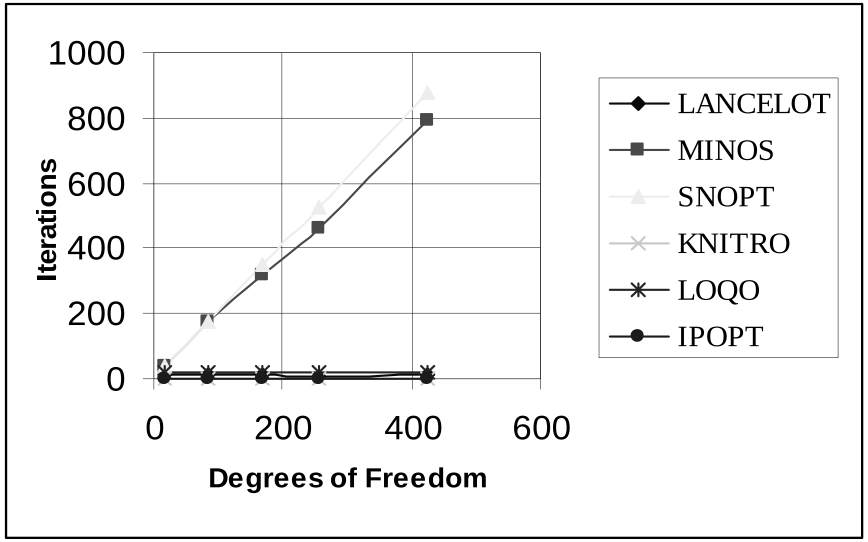
	no. of iterations	objective	CPU (s)	normalized CPU (s)
IHM Day 10 ($N = 20\,826, M = 16\,074, S = 15\,206$)				
LANCELOT	*c*	*c*	*c*	*c*
MINOS	*a*	*a*	*a*	*a*
SNOPT	*a*	*a*	*a*	*a*
KNITRO	*a*	*a*	*a*	*a*
LOQO	*b*	*b*	*b*	*b*
IPOPT, exact	65	2.64×10^4	1.12×10^4	1.12×10^4
IHM Day 15 ($N = 31\,743, M =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

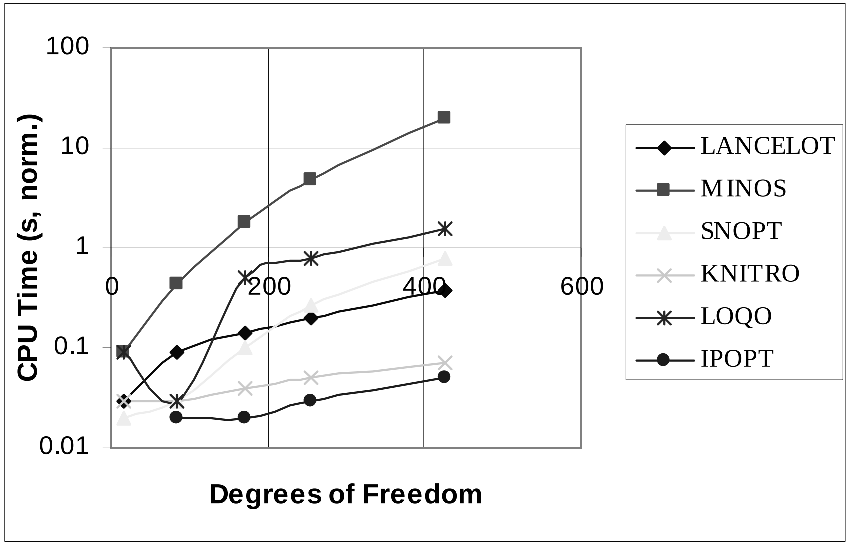
Summary of Results – Dolan-Moré plot



Comparison of NLP Solvers: Data Reconciliation







Sensitivity Analysis for Nonlinear Programming

At nominal conditions, p_0

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

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 parameteric uncertainties?
- Can this be analyzed easily?

Calculation of NLP Sensitivity

Take KKT Conditions

$$\nabla L(x^*, p, \lambda, v) = 0$$
$$c(x^*, p_0) = 0$$
$$E^T x^* - bnd(\, p_0)=0$$

and differentiate and expand about p_0.

$$\nabla_{px} L(x^*, p, \lambda, v)^T + \nabla_{xx} L(x^*, p, \lambda, v)^T \nabla_p x^{*T} + \nabla_x h(x^*, p, \lambda, v)^T \nabla_p \lambda^T + E \nabla_p v^T = 0$$
$$\nabla_p c(x^*, p_0)^T + \nabla_x c(x^*, p_0)^T \nabla_p x^{*T} = 0$$
$$E^T (\nabla_p x^{*T} - \nabla_p bnd^T)=0$$

<u>Notes:</u>

- A key assumption is that under strict complementarity, the active set will not change for small perturbations of p.
- If an element of x^* is at a bound then $\nabla_p x_i^{*T} = \nabla_p bnd^T$
- Second derivatives are required to calculate sensitivities, $\nabla_p x^{*T}$
- Is there a cheaper way to calculate $\nabla_p x^{*T}$?

Decomposition for NLP Sensitivity

Let $L(x^*, p, \lambda, v) = f(x^*) + \lambda^T c(x^*) + (E^T(x^* - bnd(p)))^T v$

$$\begin{bmatrix} W & A & E \\ A^T & 0 & 0 \\ E^T & 0 & 0 \end{bmatrix} \begin{bmatrix} \nabla_p x^{*T} \\ \nabla_p \lambda^T \\ \nabla_p v^T \end{bmatrix} = -\begin{bmatrix} \nabla_{xp} L(x^*, p, \lambda, v)^T \\ \nabla_p c(x^*, p)^T \\ -E^T \nabla_p bnd^T \end{bmatrix}$$

•Partition variables into basic, nonbasic and superbasic

$$\nabla_p x^T = Z\ \nabla_p x_S^T + Y \nabla_p x_B^T + T \nabla_p x_N^T$$

•Set $\nabla_p x_N^T = \nabla_p bnd_N^T$, nonbasic variables to rhs,

•Substitute for remaining variables

•Perform range and null space decomposition

•Solve only for $\nabla_p x_S^T$ and $\nabla_p x_B^T$

Decomposition for NLP Sensitivity

$$\begin{bmatrix} Y^TWY & Y^TWY & Y^TA \\ Z^TWY & Z^TWZ & 0 \\ A^TY & 0 & 0 \end{bmatrix}\begin{bmatrix} \nabla_p x_B{}^T \\ \nabla_p x_S{}^T \\ \nabla_p \lambda^T \end{bmatrix} = -\begin{bmatrix} Y^T(\nabla_{xp}L(x^*, p, \lambda, v)^T + W\,T\nabla_p x_N{}^T) \\ Z^T(\nabla_{xp}L(x^*, p, \lambda, v)^T + W\,T\nabla_p x_N{}^T) \\ \nabla_p c(x^*, p)^T + A^T T\nabla_p x_N{}^T \end{bmatrix}$$

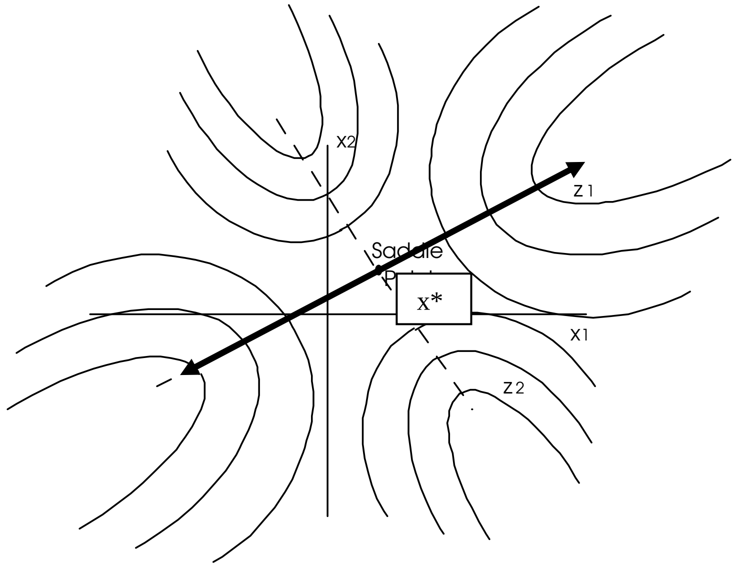
- Solve only for $\nabla_p x_B{}^T$ from bottom row and $\nabla_p x_S{}^T$ from middle row
- If second derivatives are not available, Z^TWZ, $Z^TWY\ \nabla_p x_B{}^T$ and $Z^TWT\ \nabla_p x_N{}^T$ can be constructed by directional finite differencing
- If assumption of strict complementarity is violated, sensitivity can be calculated using a QP subproblem.

Second Order Tests

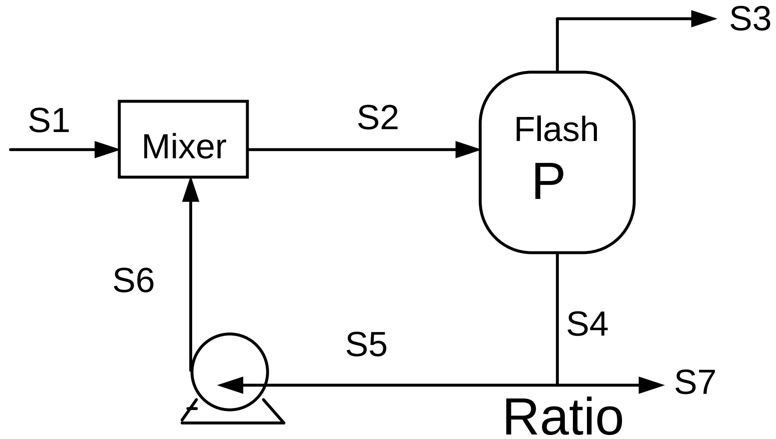
Reduced Hessian needs to be positive definite

At solution x*: Evaluate eigenvalues of $Z^T \nabla_{xx} L^* Z$
Strict local minimum if all positive.

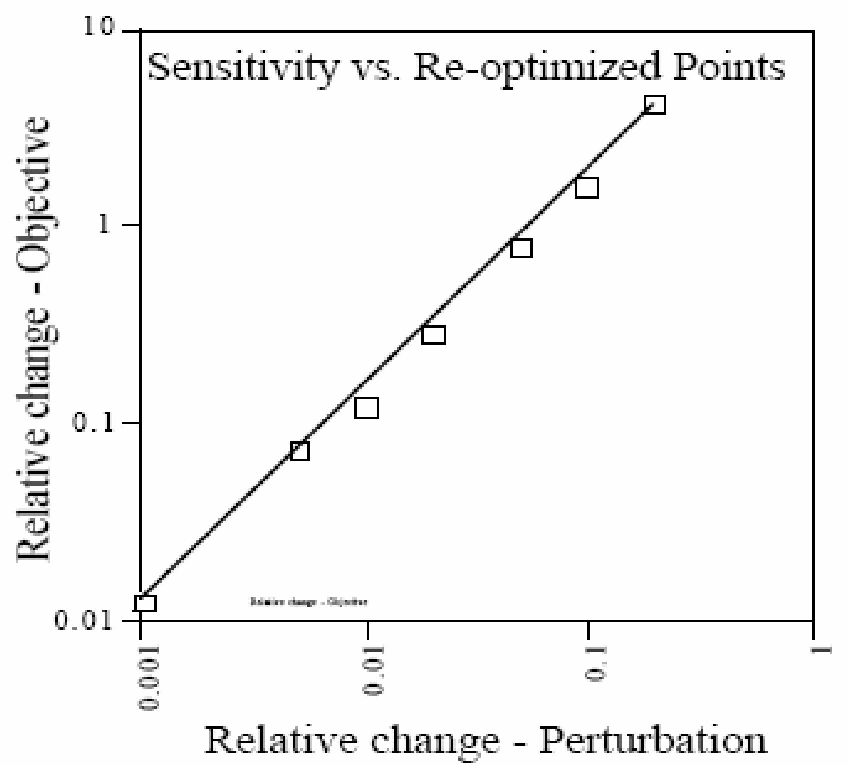
- Nonstrict local minimum: If nonnegative, find eigenvectors for zero eigenvalues, ➔ regions of nonunique solutions
- Saddle point: If any are negative, move along directions of corresponding eigenvectors and restart optimization.



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

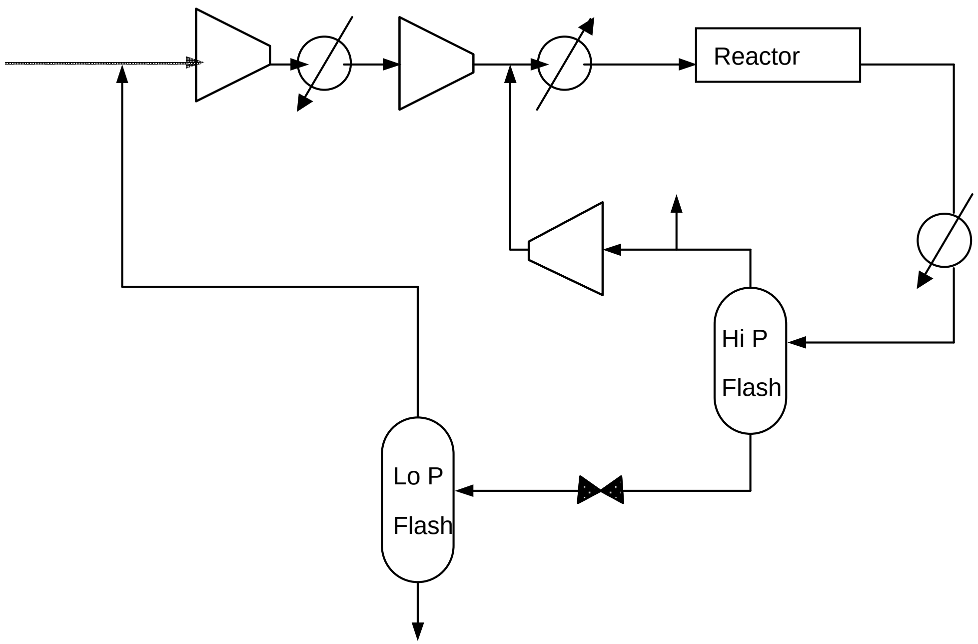


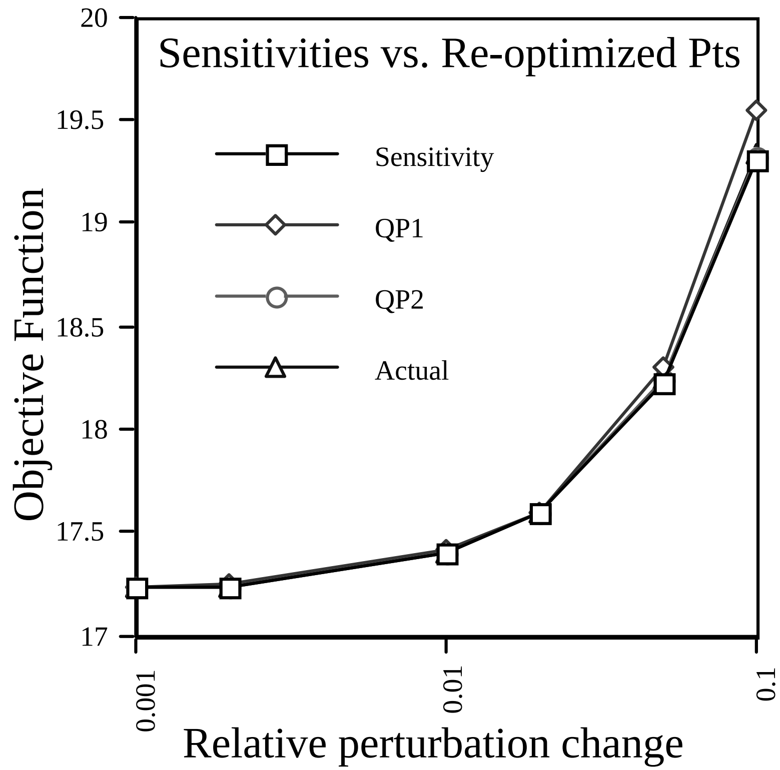
$$\text{Max } S3(A)^2 * S3(B) - S3(A)^2 - S3(C)^3 + S3(D) - (S3(E))^{1/2}$$



- Second order sufficiency test:
- Dimension of reduced Hessian = 1
- Positive eigenvalue
- Sensitivity to simultaneous change in feed rate and upper bound on purge ratio
- Only 2-3 flowsheet perturbations required for second order information

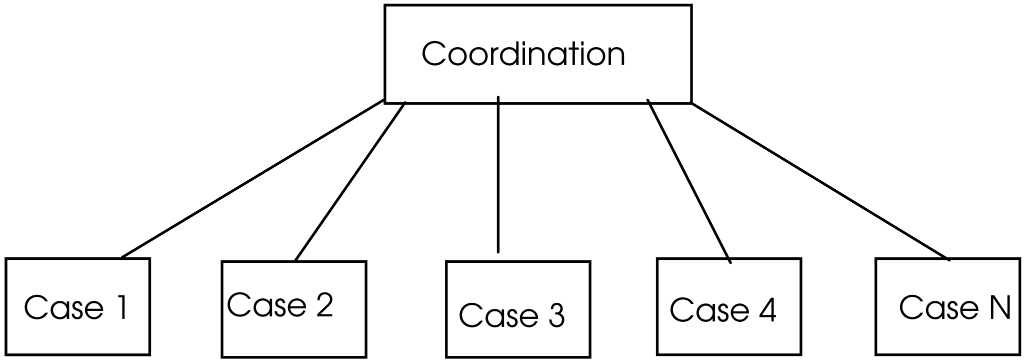
Ammonia Process Optimization (9 decisions, 8 tear variables)





- 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
- Only 5-6 extra perturbations for second derivatives

Multiperiod Optimization

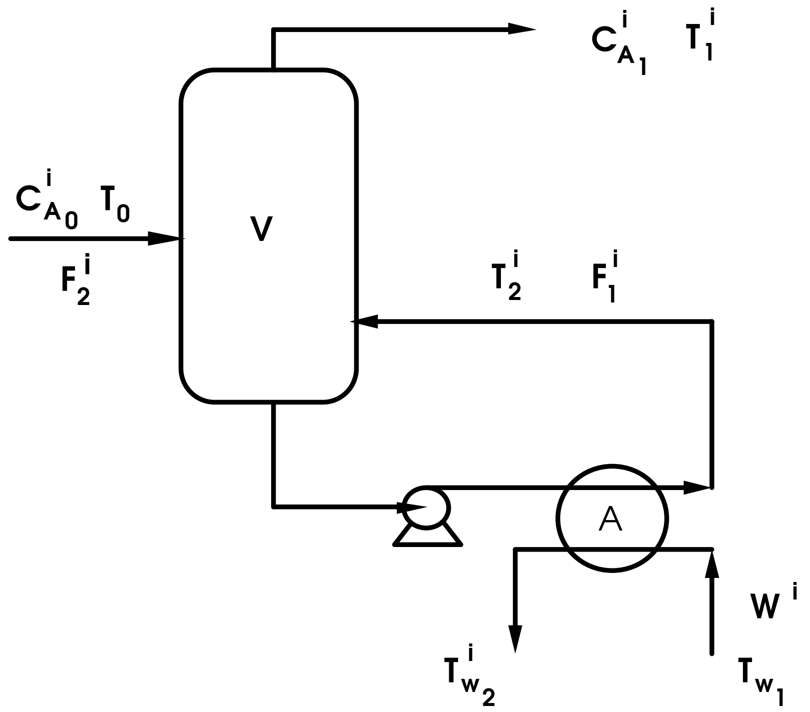


1. Design plant to deal with different operating scenarios (over time or with uncertainty)

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

Multiperiod Flowsheet Example



Parameters	Period 1	Period 2	Period 3	Period 4
E (kJ/mol)	555.6	583.3	611.1	527.8
k_0 (1/h)	10	11	12	9
F (kmol/h)	45.4	40.8	24.1	32.6
Time (h)	1000	4000	2000	1000

Multiperiod Design Model

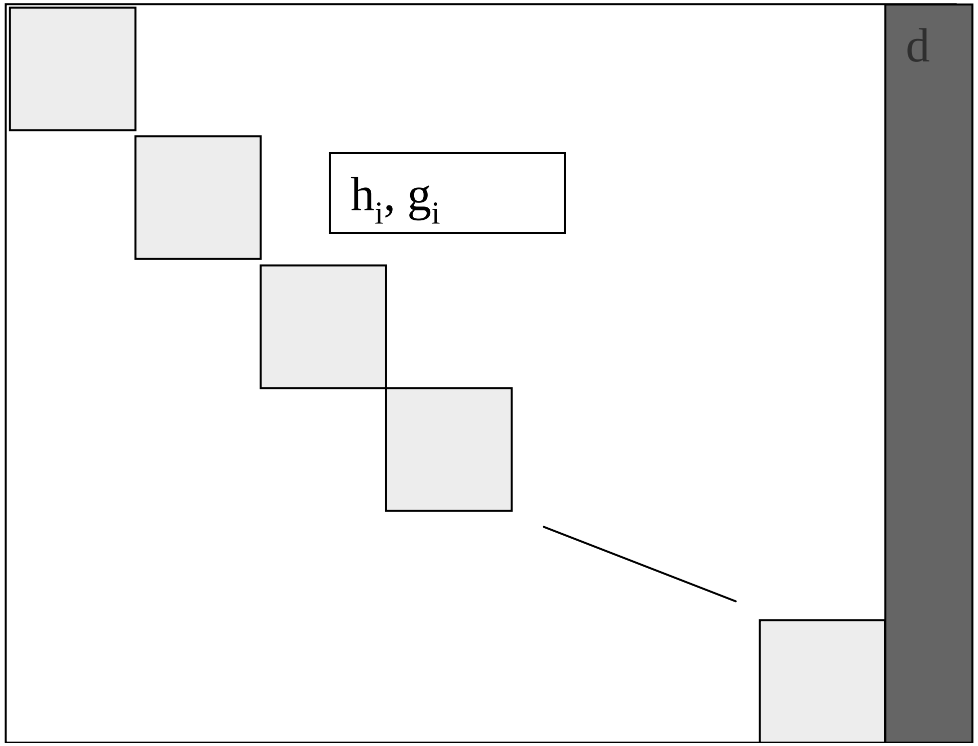
$$Min\ f_0(d) + \Sigma_i f_i(d, x_i)$$
$$s.t.\ h_i(x_i, d) = 0,\ i = 1,\dots N$$
$$g_i(x_i, d) \le 0,\ i = 1,\dots N$$
$$r(d) \le 0$$

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 δ_i = d



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

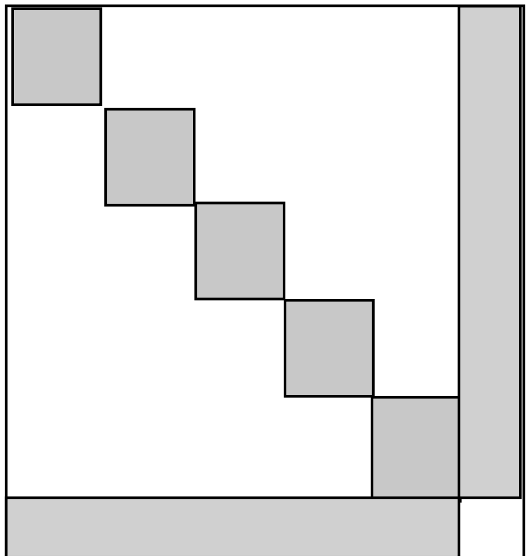
Multiperiod Decomposition Strategy
SQP Subproblem

$$\text{minimize } \phi = \nabla_{\mathrm{d}} f_0^T \, \Delta d + \frac{1}{2} \Delta d^T \, \nabla_{\mathrm{d}}^2 L_0 \, \Delta d + \sum_{\mathrm{i=1}}^{\mathrm{N}} \left(\nabla_{\mathrm{p}} f_i^T \, p_i + \frac{1}{2} p_i^T \, \nabla_{\mathrm{p}}^2 L_i \, p_{\mathrm{i}} \right)$$

$$\text{subject to} \quad \overline{h}_i + \nabla_p \overline{h}_i^T \, p_i = 0 \qquad i = 1, \ldots N$$

$$r + \nabla_d r^T \, \Delta d \leq 0$$

$$p_i = \begin{bmatrix} x_i^{k+1} - x_i^k \\ s_i^{k+1} - s_i^k \\ \delta_i^{k+1} - \delta_i^k \end{bmatrix} \qquad \overline{h}_i = \begin{bmatrix} h_i^k \\ g_i^k + s_i^k \\ -\Delta d \end{bmatrix} \qquad \nabla_p \overline{h}_i = \begin{bmatrix} & | & & | \; 0 \\ \nabla_p h_i^k & | & \nabla_p (g_i^k + s_i^k) & | \; 0 \\ & | & & | \; \mathrm{I} \end{bmatrix}$$



- Block diagonal bordered KKT matrix (arrowhead structure)
- Solve each block sequentially (range/null dec.) to form small QP in space of d variables
- Reassemble all other steps from QP solution

Multiperiod Decomposition Strategy

From decomposition of KKT block in each period, obtain the following directions that are parametric in Δd:

$$p_{Z_i} = A_{Z_i} + B_{Z_i}\,\Delta d \qquad \text{and} \qquad Z_i\, p_{Z_i} = Z_{A_i} + Z_{B_i}\,\Delta d$$

$$p_{Y_i} = A_{Y_i} + B_{Y_i}\,\Delta d \qquad \text{and} \qquad Y_i\, p_{Y_i} = Y_{A_i} + Y_{B_i}\,\Delta d$$

Substituting back into the original QP subproblem leads to a QP only in terms of Δd.

$$\text{minimize } \phi = \left[\nabla_{\mathrm{d}} f_0^T + \sum_{i=1}^{N}\{\nabla_{\mathrm{p}} f_i^T (Z_{B_i} + Y_{B_i}) + (Z_{A_i} + Y_{A_i})^T \nabla_{\mathrm{p}}^2 L_i (Z_{B_i} + Y_{B_i})\}\right]\Delta d$$

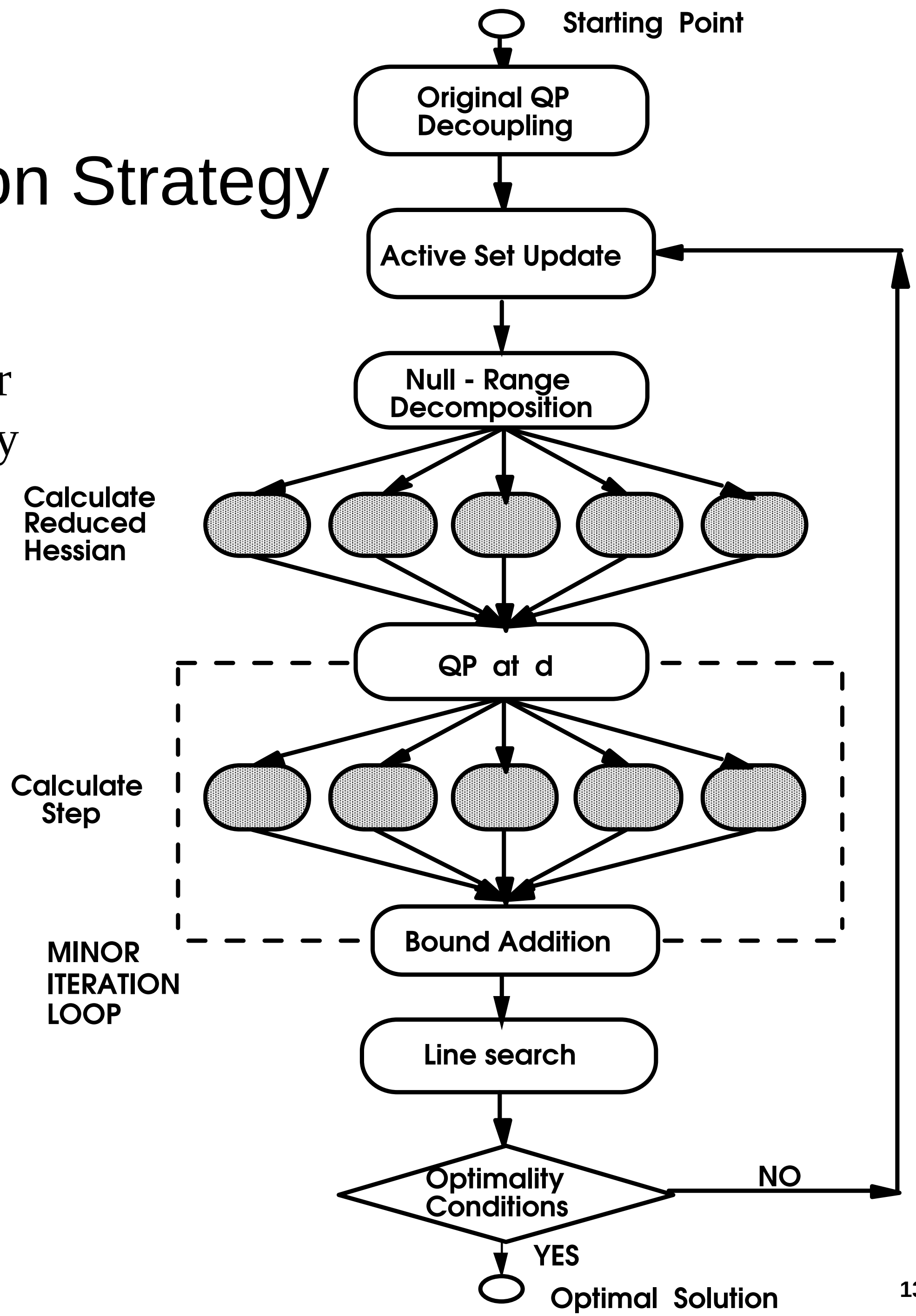
$$+ \frac{1}{2}\Delta d^T \left[\nabla_{\mathrm{d}}^2 L_0 + \sum_{i=1}^{N}\{(Z_{B_i} + Y_{B_i})^T \nabla_{\mathrm{p}}^2 L_i (Z_{B_i} + Y_{B_i})\}\right]\Delta d$$

$$\text{subject to} \qquad r + \nabla_{\mathrm{d}} r\,\Delta d \le 0$$

Once Δd is obtained, directions are obtained from the above equations.

Multiperiod Decomposition Strategy

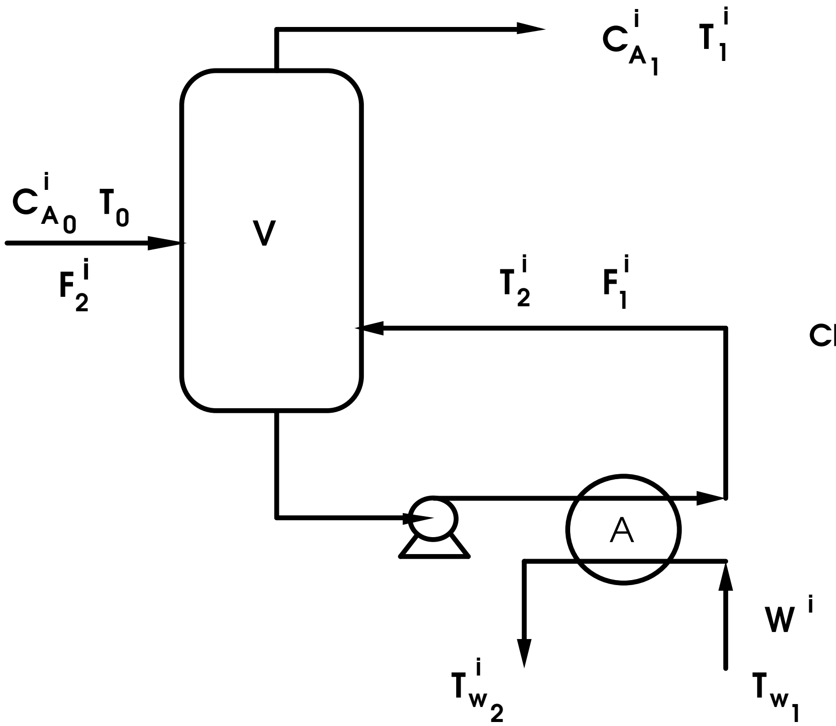
- p_i steps are parametric in Δd and their components are created independently
- Decomposition linear in number of periods and trivially parallelizable
- Choosing the active inequality constraints can be done through:
  - -Active set strategy (e.g., bound addition)
  - -Interior point strategy using barrier terms in objective
- Easy to implement in decomposition

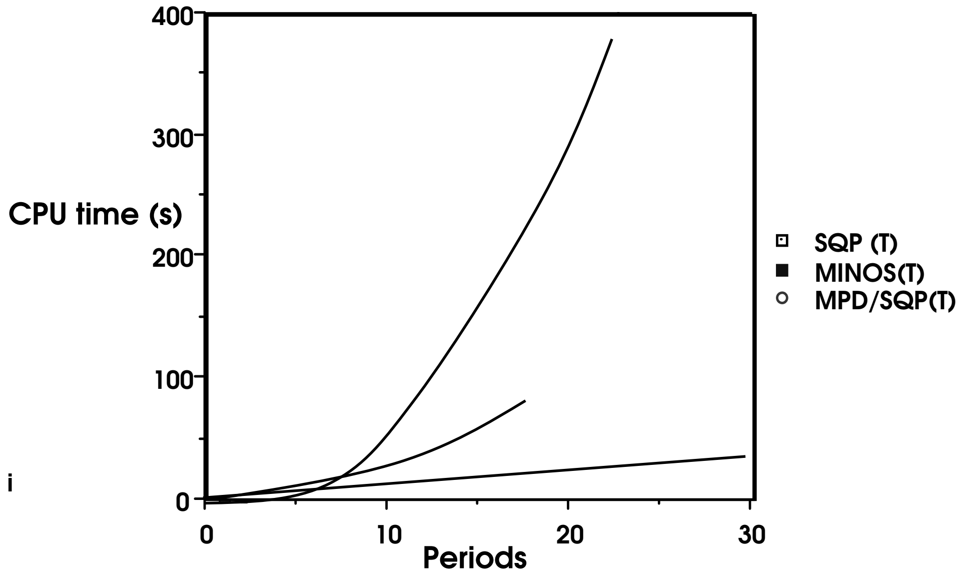


Multiperiod Flowsheet 1

(13+2) variables and (31+4) constraints (1 period)

262 variables and 624 constraints (20 periods)

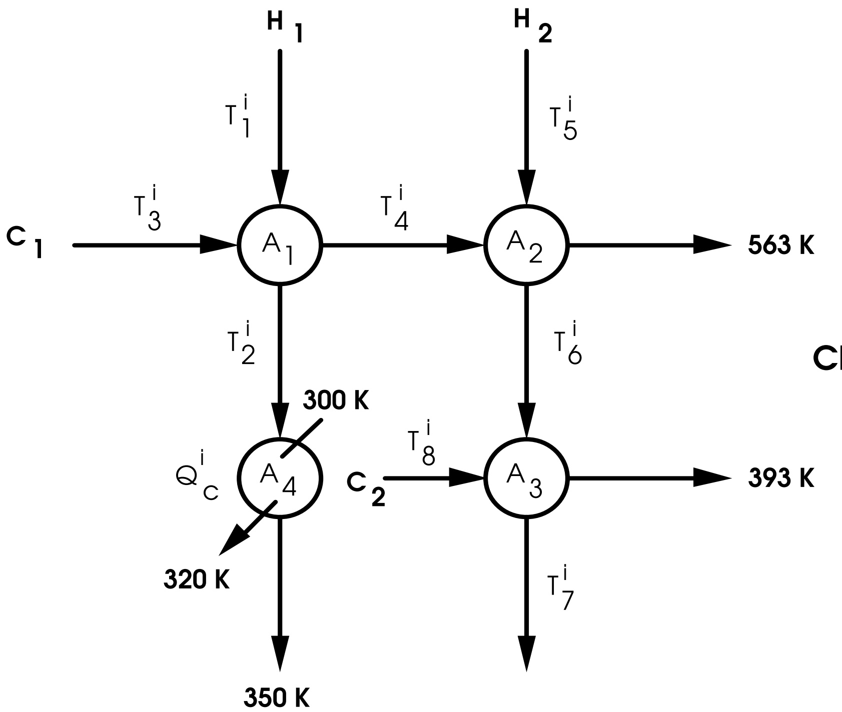


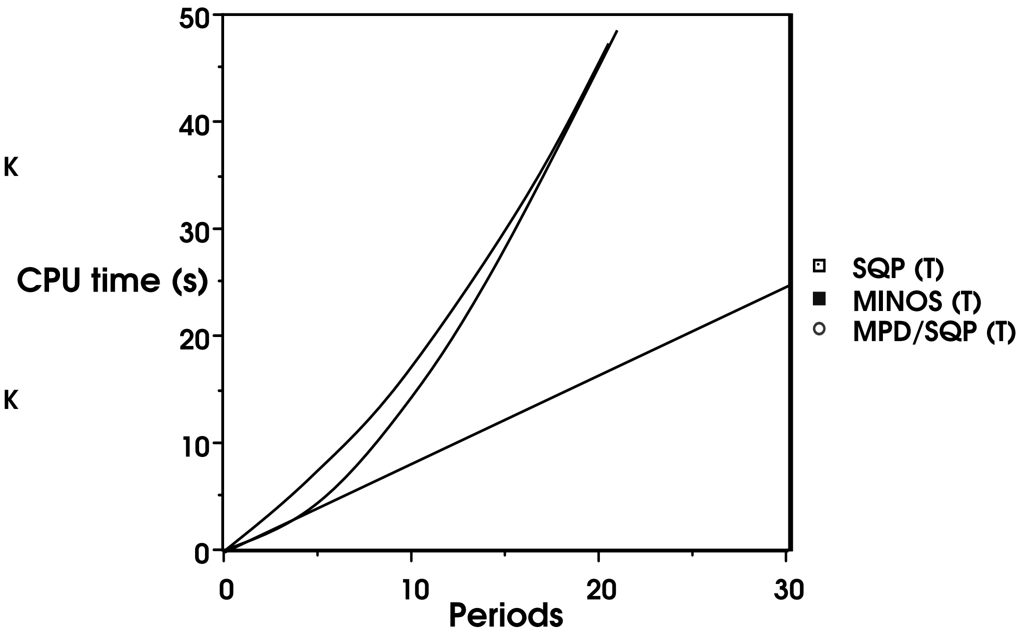


Multiperiod Example 2 – Heat Exchanger Network

(12+3) variables and (31+6) constraints (1 period)

243 variables and 626 constraints (20 periods)





Summary and Conclusions

-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
-Multiperiod Optimization Problems

Optimization of Differential-Algebraic Equation Systems

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

DAE Optimization Outline

I Introduction
- Process Examples

II Parametric Optimization
- Gradient Methods
  - Perturbation
  - Direct - Sensitivity Equations
  - Adjoint Equations

III Optimal Control Problems
- Optimality Conditions
- Model Algorithms
  - Sequential Methods
  - Multiple Shooting
  - Indirect Methods

IV Simultaneous Solution Strategies
- Formulation and Properties
- Process Case Studies
- Software Demonstration

Dynamic Optimization Problem

$$\min \quad \Phi\big(z(t),\ y(t),\ u(t),\ p,\ t_f\big)$$

s.t.

$$\frac{dz(t)}{dt} = f\big(z(t), y(t), u(t), t, p\big)$$

$$g\big(z(t), y(t), u(t), t, p\big) = 0$$

$$z^o = z(0)$$

$$z^l \le z(t) \le z^u$$

$$y^l \le y(t) \le y^u$$

$$u^l \le u(t) \le u^u$$

$$p^l \le p \le p^u$$

t, time
z, differential variables
y, algebraic variables

t_f, final time
u, control variables
p, time independent parameters

DAE Models in Process Engineering

Differential Equations

- Conservation Laws (Mass, Energy, Momentum)

Algebraic Equations

- Constitutive Equations, Equilibrium (physical properties, hydraulics, rate laws)
- Semi-explicit form
- Assume to be index one (i.e., algebraic variables can be solved uniquely by algebraic equations)
- If not, DAE can be reformulated to index one (see Ascher and Petzold)

Characteristics

- Large-scale models – not easily scaled
- Sparse but no regular structure
- Direct linear solvers widely used
- Coarse-grained decomposition of linear algebra

Parameter Estimation

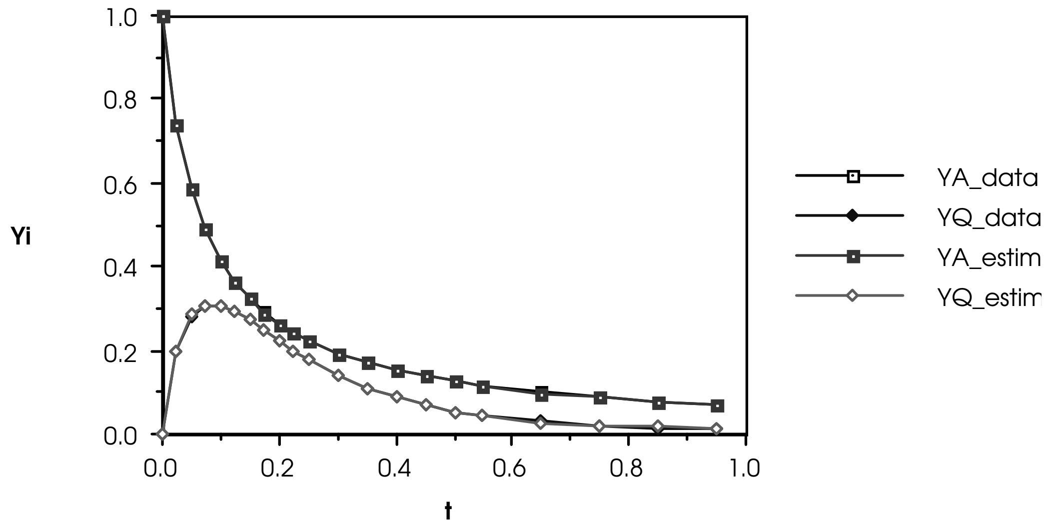
Catalytic Cracking of Gasoil (Tjoa, 1991)

$$A \xrightarrow{p_1} Q,\ Q \xrightarrow{p_2} S,\ A \xrightarrow{p_3} S$$

$$\dot{a} = -(p_1 + p_3)a^2$$

$$\dot{q} = -p_1 a^2 - p_2 q$$

$$a(0) = 1, \quad q(0) = 0$$



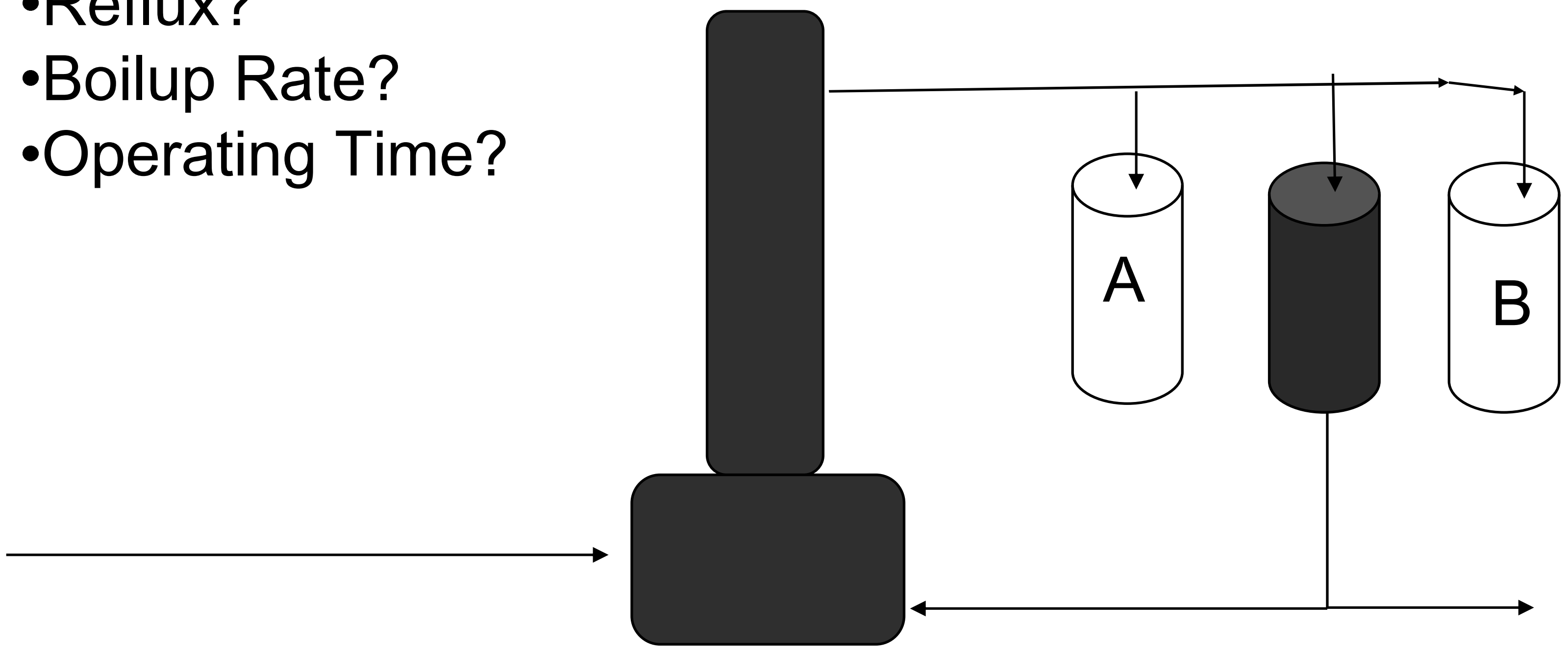
number of states and ODEs: 2
number of parameters:3
no control profiles
constraints: $p_L \leq p \leq p_U$

Objective Function: Ordinary Least Squares

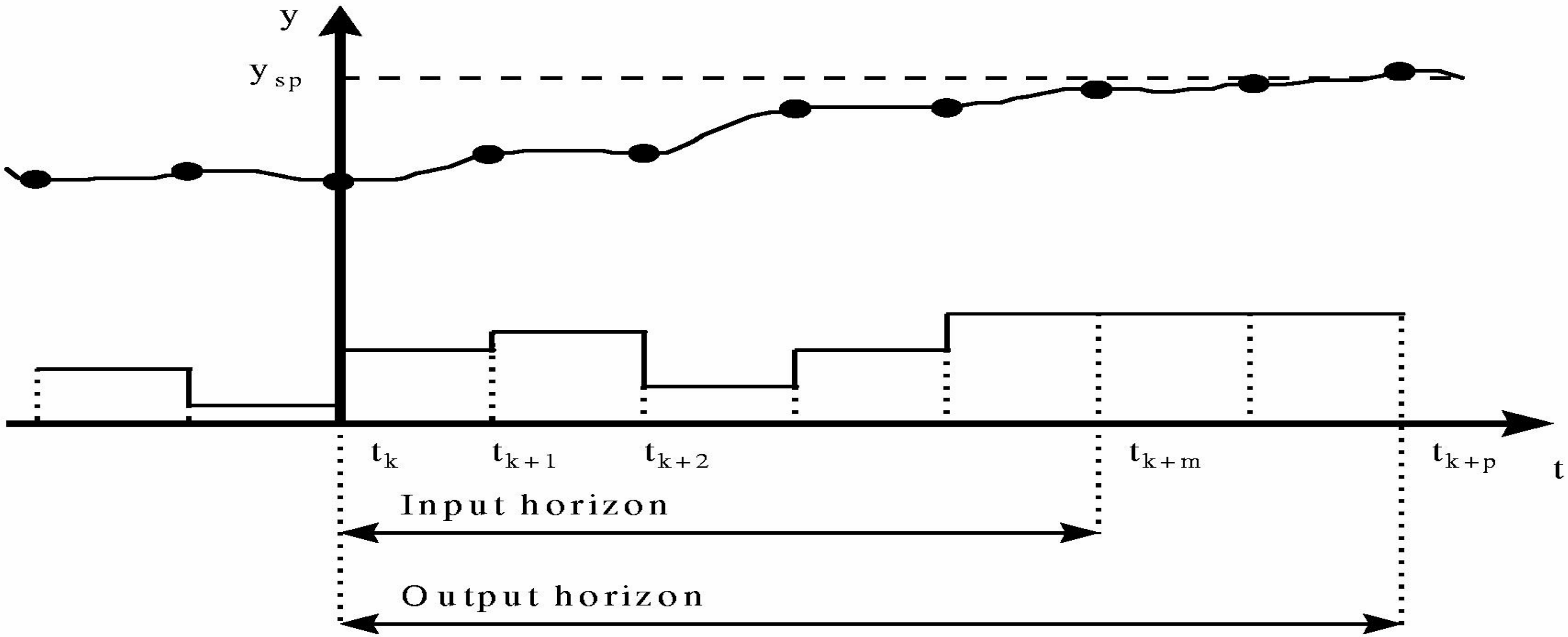
$(p_1, p_2, p_3)^0 = (6, 4, 1)$
$(p_1, p_2, p_3)^* = (11.95, 7.99, 2.02)$
$(p_1, p_2, p_3)_{true} = (12, 8, 2)$

Batch Distillation Multi-product Operating Policies

- Run between distillation batches
- Treat as boundary value optimization problem
  - When to switch from A to offcut to B?
  - How much offcut to recycle?
  - Reflux?
  - Boilup Rate?
  - Operating Time?



Nonlinear Model Predictive Control (NMPC)



$$\min_{\boldsymbol{u}} \quad \sum \| y(t) - y^{\mathrm{sp}} \|_{Q^y} + \sum \| \boldsymbol{u}(\boldsymbol{t}^k) - \boldsymbol{u}(\boldsymbol{t}^{k-1}) \|_{Q^u}$$

$$s.t. \quad \begin{aligned} & z'(t) = F(z(t), y(t), \boldsymbol{u}(t), t) \\ & 0 = G(z(t), y(t), \boldsymbol{u}(t), t) \\ & z(t) = z^{\mathrm{init}} \\ & \text{Bound Constraints} \\ & \text{Other Constraints} \end{aligned}$$

Batch Process Optimization

Optimization of dynamic batch process operation resulting from reactor and distillation column

DAE models:

$z' = f(z, y, u, p)$

$g(z, y, u, p) = 0$

$A+B \rightarrow C$

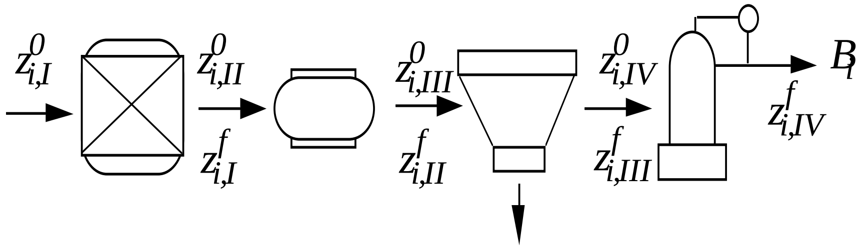
$C+B \rightarrow P+E$

$P+C \rightarrow G$

number of states and DAEs: $n_z + n_y$

parameters for equipment design (reactor, column)

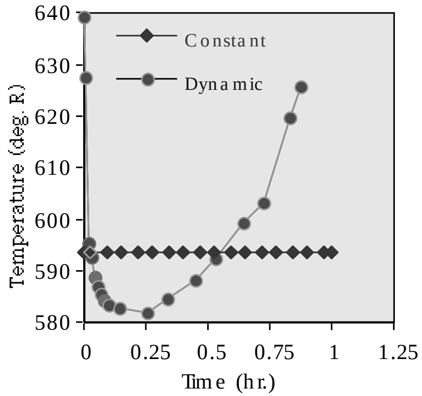
n_u control profiles for optimal operation



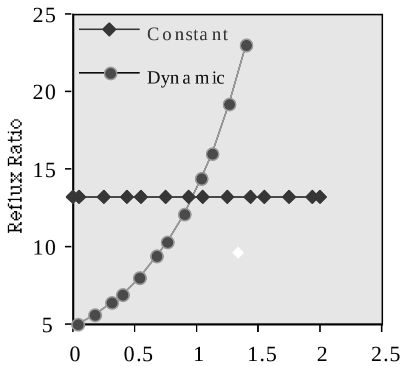
Constraints: $u_L \leq u(t) \leq u_U$ $\quad z_L \leq z(t) \leq z_U$

$y_L \leq y(t) \leq y_U$ $\quad p_L \leq p \leq p_U$

Objective Function: amortized economic function at end of cycle time t_f



optimal reactor temperature policy



optimal column reflux ratio

Reactor Design Example

Plug Flow Reactor Optimization

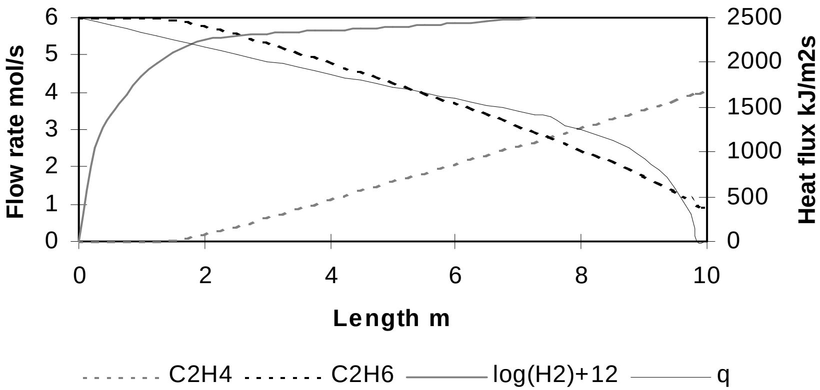
The cracking furnace is an important example in the olefin production industry, where various hydrocarbon feedstocks react. Consider a simplified model for ethane cracking (Chen et al., 1996). The objective is to find an optimal profile for the heat flux along the reactor in order to maximize the production of ethylene.

$$\begin{aligned} &\text{Max } F_{exit}^{C_2H_4} \\ &\text{s.t. DAE} \\ &T_{exit} \leq 1180\text{K} \end{aligned}$$

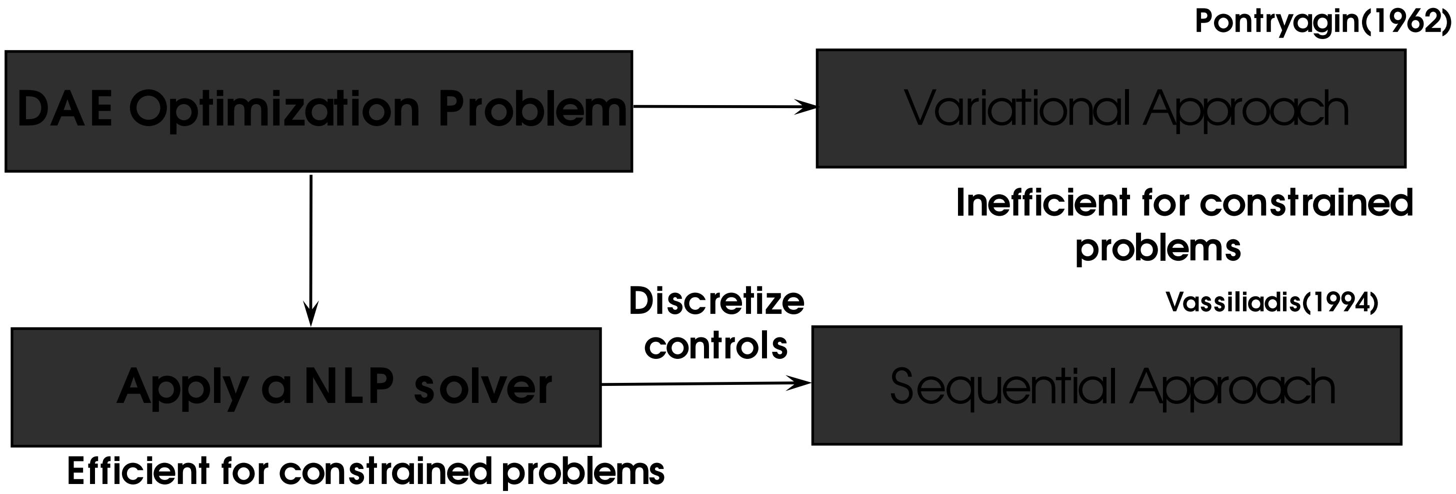
The reaction system includes six molecules, three free radicals, and seven reactions. The model also includes the heat balance and the pressure drop equation. This gives a total of eleven differential equations.

$$C_2H_6 \rightarrow 2CH_3\bullet$$
$$H\bullet + C_2H_6 \rightarrow H_2 + C_2H_5\bullet$$
$$C_2H_5\bullet + C_2H_4 \rightarrow C_3H_6 + CH_3\bullet$$
$$H\bullet + C_2H_4 \rightarrow C_2H_5\bullet$$
$$CH_3\bullet + C_2H_6 \rightarrow CH_4 + C_2H_5\bullet$$
$$C_2H_5\bullet \rightarrow C_2H_4 + H\bullet$$
$$2C_2H_5\bullet \rightarrow C_4H_{10}$$

Concentration and Heat Addition Profile



Dynamic Optimization Approaches



Sequential Approaches - Parameter Optimization

Consider a simpler problem without control profiles:

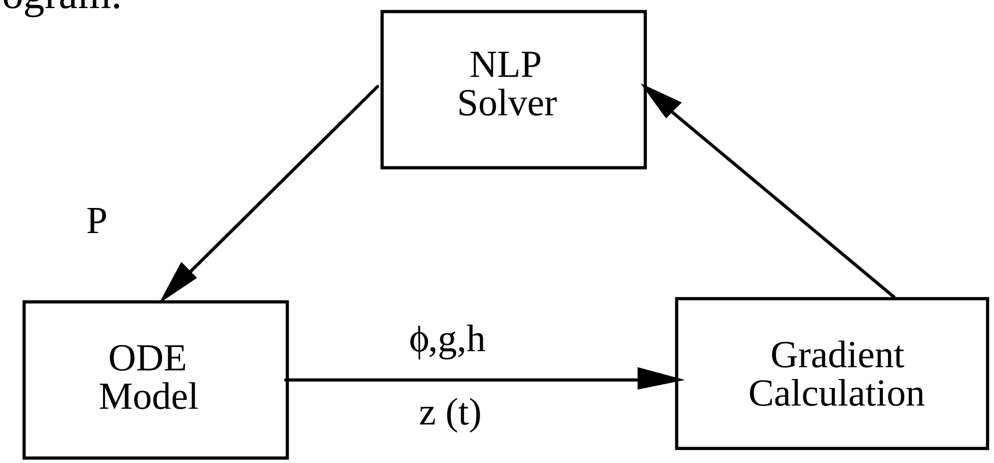
e.g., equipment design with DAE models - reactors, absorbers, heat exchangers

$$Min \quad \Phi(z(t_f))$$

$$z' = f(z, p),\ z(0) = z_0$$

$$g(z(t_f)) \le 0,\ h(z(t_f)) = 0$$

By treating the ODE model as a "black-box" a sequential algorithm can be constructed that can be treated as a nonlinear program.



Task: How are gradients calculated for optimizer?

Gradient Calculation

Perturbation

Sensitivity Equations

Adjoint Equations

Perturbation

Calculate approximate gradient by solving ODE model (np + 1) times

Let $\psi = \Phi$, g and h (at $t = t_f$)

$$d\psi/dp_i = \{\psi\,(p_i + \Delta p_i) - \psi\,(p_i)\}/\,\Delta p_i$$

Very simple to set up

Leads to poor performance of optimizer and poor detection of optimum unless roundoff error ($O(1/\Delta p_i)$) and truncation error ($O(\Delta p_i)$) are small.

Work is proportional to np (expensive)

Direct Sensitivity

From ODE model: $$\frac{\partial}{\partial p}\left\{z' = f(z, p, t), z(0) = z_0(p)\right\}$$

$$\text{define } s_i(t) = \frac{\partial z(t)}{\partial p_i} \; i = 1, \ldots \text{np}$$

$$s_i' = \frac{d}{dt}(s_i) = \frac{\partial f}{\partial p_i} + \frac{\partial f}{\partial z}^T s_i, \; s_i(0) = \frac{\partial z(0)}{\partial p_i}$$

(nz x np sensitivity equations)

• z and s_i , i = 1,…np, an be integrated forward simultaneously.

• for implicit ODE solvers, $s_i(t)$ can be carried forward in time after converging on z

• linear sensitivity equations exploited in ODESSA, DASSAC, DASPK, DSL48s and a number of other DAE solvers

Sensitivity equations are efficient for problems with many more constraints than parameters (1 + ng + nh > np)

Example: Sensitivity Equations

$$z_1' = z_1^2 + z_2^2$$

$$z_2' = z_1 z_2 + z_1 p_b$$

$$z_1 = 5, z_2(0) = p_a$$

$$s(t)_{a,j} = \partial z(t)_j / \partial p_a, s(t)_{b,j} = \partial z(t)_j / \partial p_b, j = 1,2$$

$$s_{a,1}' = 2z_1 s_{a,1} + 2z_2 s_{a,2}$$

$$s_{a,2}' = z_1 s_{a,2} + z_2 s_{a,1} + s_{a,1} p_b$$

$$s_{a,1} = 0, s_{a,2}(0) = 1$$

$$s_{b,1}' = 2z_1 s_{b,1} + 2z_2 s_{b,2}$$

$$s_{b,2}' = z_1 + z_1 s_{b,2} + z_2 s_{b,1} + s_{b,1} p_b$$

$$s_{b,1} = 0, s_{b,2}(0) = 0$$

Adjoint Sensitivity

Adjoint or Dual approach to sensitivity

Adjoin model to objective function or constraint

($\psi = \Phi$,g or h)

$$\psi = \psi(t_f) - \int_0^{t_f} \lambda^T (z' - f(z, p, t))dt$$

$$\psi = \psi(t_f) + \lambda(0)^T z_0(p) - \lambda(t_f)^T z(t_f) + \int_0^{t_f} z^T \lambda' + \lambda^T F(z, p, t))dt$$

(λ(t)) serve as multipliers on ODE's)

Now, integrate by parts

$$d\psi = \cancelto{0}{\left[\frac{\partial \psi(z(t_f))}{\partial z(t_f)} - \lambda(t_f)\right]} \delta z(t_f) + \left[\frac{\partial z_0(p)}{\partial p}\lambda(0)\right]^T dp + \int_0^{t_f} \cancelto{0}{\left[\lambda' + \frac{\partial f}{\partial z}\lambda\right]^T} \delta z(t) + \left[\frac{\partial f}{\partial p}\lambda\right]^T dp\; dt$$

and find $d\psi/dp$ subject to feasibility of ODE's

Now, set all terms not in dp to zero.

Adjoint System

$$\lambda' = -\frac{\partial f}{\partial z}\lambda(t), \;\; \lambda(t_f) = \frac{\partial \psi(z(t_f))}{\partial z(t_f)}$$

$$\frac{d\psi}{dp} = \frac{\partial z_0(p)}{\partial p}\lambda(0) + \int_0^{t_f}\left[\frac{\partial f}{\partial p}\lambda(t)\right]dt$$

Integrate model equations <u>forward</u>

Integrate adjoint equations backward and evaluate integral and sensitivities.

Notes:

nz (ng + nh + 1) adjoint equations must be solved backward (one for each objective and constraint function)

for implicit ODE solvers, profiles (and even matrices) can be stored and carried backward after solving forward for z as in DASPK/Adjoint (Li and Petzold)

more efficient on problems where: np > 1 + ng + nh

Example: Adjoint Equations

$$z_1' = z_1^2 + z_2^2$$
$$z_1' = z_1 z_2 + z_1 p_b$$
$$z_1 = 5, z_2(0) = p_a$$

Form $\lambda^T f(z,p,t) = \lambda_1(z_1^2 + z_2^2) + \lambda_2(z_1 z_2 + z_1 p_b)$

$$\lambda' = -\frac{\partial f}{\partial z}\lambda(t), \;\; \lambda(t_f) = \frac{\partial \psi(z(t_f))}{\partial z(t_f)}$$

$$\frac{d\psi}{dp} = \frac{\partial z_0(p)}{\partial p}\lambda(0) + \int_0^{t_f}\left[\frac{\partial f}{\partial p}\lambda(t)\right]dt$$

then becomes:

$$\lambda_1' = -2\lambda_1 z_1 - \lambda_2(z_2 + p_b), \;\; \lambda_1(t_f) = \frac{\partial \psi(t_f)}{\partial z_1(t_f)}$$

$$\lambda_2' = -2\lambda_1 z_2 - \lambda_2 z_1, \;\; \lambda_2(t_f) = \frac{\partial \psi(t_f)}{\partial z_2(t_f)}$$

$$\frac{d\psi(t_f)}{dp_a} = \lambda_1(0)$$

$$\frac{d\psi(t_f)}{dp_b} = \int_0^{t_f} \lambda_2(t) z_1(t)\, dt$$

Example: Hot Spot Reactor

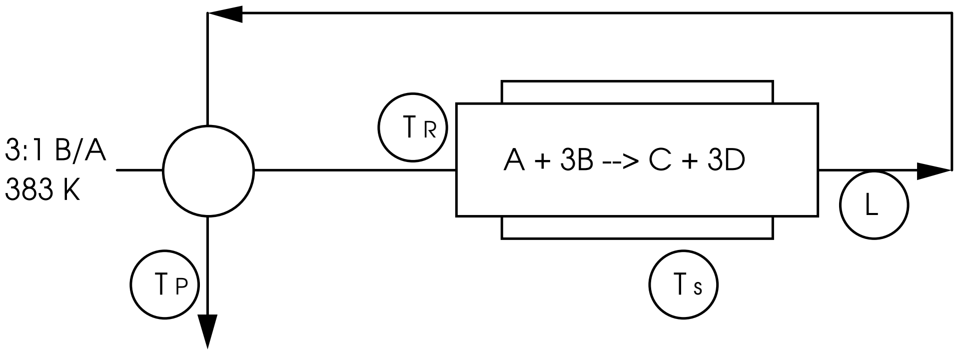
$$\underset{T_P, T_R, L, T_S}{Min} \quad \Phi = L - \int_0^L (T(t) - T_S / T_R) dt$$

$$s.t. \quad \frac{dq}{dt} = 0.3(1 - q(t)) \exp[20 - 20/T(t)], \; q(0) = 0$$

$$\frac{dT}{dt} = -1.5(T(t) - T_S / T_R) + 2/3 \frac{dq}{dt}, \; T(0) = 1$$

$$\Delta H_{feed}(T_R, 110^o C) - \Delta H_{product}(T_P, T(L)) = 0$$

$$T_P = 120^o C, \; T(L) = 1 + 10^o C/T_R$$



T_P = specified product temperature
T_R = reactor inlet, reference temperature
L = reactor length
T_S = steam sink temperature
q(t) = reactor conversion profile
T(t) = normalized reactor temperature profile

Cases considered:

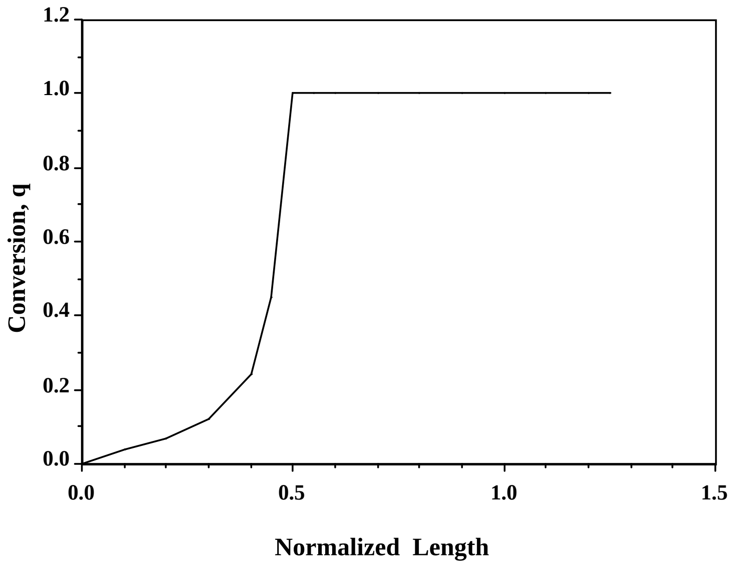
- Hot Spot - no state variable constraints
- Hot Spot with $T(t) \leq 1.45$

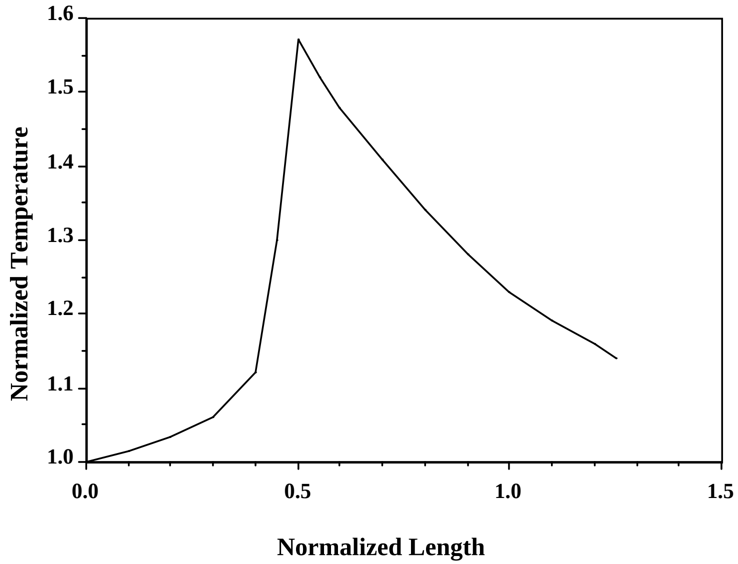
Hot Spot Reactor: Unconstrained Case

Method: SQP (perturbation derivatives)

	L(norm)	T_R(K)	T_S(K)	T_P(K)
Initial:	1.0	462.23	425.26	250
Optimal:	1.25	500	470.1	188.4

13 SQP iterations / 2.67 CPU min. (μVax II)





Constrained Temperature Case: could not be solved with sequential method

Tricks to generalize classes of problems

<u>Variable Final Time</u> (Miele, 1980)

Define $t = p_{n+1}\,\tau,\ 0 \le \tau \le 1,\ p_{n+1} = t_f$

Let $dz/dt = (1/p_{n+1})\, dz/d\tau = f(z, p) \Rightarrow dz/d\tau = (p_{n+1})\, f(z, p)$

<u>Converting Path Constraints to Final Time</u>

Define measure of infeasibility as a new variable, $z_{nz+1}(t)$ (Sargent & Sullivan, 1977):

$$z_{nz+1}(t_f) = \sum_j \int_0^{t_f} \max(0, g_j(z(t), u(t))^2\, dt$$

$$or\ \ \dot{z}_{nz+1}(t) = \sum_j \max(0, g_j(z(t), u(t))^2\ ,\ z_{nz+1}(0) = 0$$

$$\text{Enforce}\quad z_{nz+1}(t_f) \le \varepsilon \quad \text{(however, constraint is degenerate)}$$

Profile Optimization - (Optimal Control)

Optimal Feed Strategy (Schedule) in Batch Reactor

Optimal Startup and Shutdown Policy

Optimal Control of Transients and Upsets

Sequential Approach: Approximate control profile as through parameters (piecewise constant, linear, polynomial, etc.)

Apply NLP to discretization as with parametric optimization

Obtain gradients through adjoints (Hasdorff; Sargent and Sullivan; Goh and Teo) or sensitivity equations (Vassiliadis, Pantelides and Sargent; Gill, Petzold et al.)

Variational (Indirect) approach: Apply optimality conditions and solve as boundary value problem

Derivation of Variational Conditions
Indirect Approach

Optimality Conditions (Bound constraints on u(t))

$$Min \quad \phi(z(tf))$$
$$s.t. \quad dz/dt = f(z, u), \; z(0) = z_0$$
$$g(z(t_f)) \le 0$$
$$h(z(t_f)) = 0$$
$$a \le u(t) \le b$$

Form Lagrange function - adjoin objective function and constraints:

$$\phi = \phi(t_f) + g(z(t_f))^T \mu + h(z(t_f))^T \nu$$
$$- \int_0^{t_f} \lambda^T (\dot{z} - f(z,u)) + \alpha_a^T (a - u(t)) + \alpha_b^T (u(t) - b)\, dt$$

Integrate by parts :

$$\phi = \phi(t_f) + g(z(t_f))^T \mu + h(z(t_f))^T \nu + \lambda^T(0) z(0) - \lambda^T(t_f) z(t_f)$$
$$+ \int_0^{t_f} \dot{\lambda}^T z + \lambda^T f(z,u) + \alpha_a^T (a - u(t)) + \alpha_b^T (u(t) - b)\, dt$$

Derivation of Variational Conditions

$$\delta\phi = \left[\frac{\partial\phi}{\partial z} + \frac{\partial g}{\partial z}\mu + \frac{\partial h}{\partial z}\nu - \lambda\right]^T \delta z(t_f) + \lambda^T(0)\delta z(0)$$

$$+\int_0^{t_f}\left[\dot{\lambda} + \frac{\partial f(z,u)}{\partial z}\lambda\right]^T \delta z(t) + \left[\frac{\partial f(z,u)}{\partial u}\lambda + \alpha_b - \alpha_a\right]^T \delta u(t)\, dt \geq 0$$

At optimum, $\delta\phi \geq 0$. Since u is the control variable, let all other terms vanish.

$\Rightarrow$ $\delta z(t_f)$:
$$\lambda(t_f) = \left\{\frac{\partial\phi}{\partial z} + \frac{\partial g}{\partial z}\mu + \frac{\partial h}{\partial z}\gamma\right\}_{t=t_f}$$

$\delta z(0)$: $\lambda(0) = 0$ (if z(0) is not specified)

$\delta z(t)$:
$$\dot{\lambda} = -\frac{\partial H}{\partial z} = -\frac{\partial f}{\partial z}\lambda$$

Define Hamiltonian, $H = \lambda^T f(z,u)$

For u <u>not</u> at bound:
$$\frac{\partial f}{\partial u}\lambda = \frac{\partial H}{\partial u} = 0$$

For u <u>at</u> bounds:
$$\frac{\partial H}{\partial u} = \alpha_a - \alpha_b$$

$$\alpha_a^T(a - u(t))$$
$$\alpha_b^T(u(t) - b)$$
$$u_a \leq u(t) \leq u_b$$
$$\alpha_a \geq 0, \alpha_b \geq 0$$

<u>Upper bound</u>, u(t) = b, $\frac{\partial H}{\partial u} = -\alpha_b \leq 0$

<u>Lower bound</u>, u(t) = a, $\frac{\partial H}{\partial u} = \alpha_a \geq 0$

Car Problem

Travel a fixed distance (rest-to-rest) in minimum time.

$$
\begin{aligned}
&Min \quad t_f \\
&s.t. \;\; x'' = u \\
&a \le u(t) \le b \\
&x(0) = 0, x(t_f) = L \\
&x'(0) = 0, x'(t_f) = 0
\end{aligned}
$$

$$
\begin{aligned}
&Min \quad x_3(t_f) \\
&s.t. \;\; x_1' = x_2 \\
&\qquad x_2' = u \\
&\qquad x_3' = 1 \\
&a \le u(t) \le b \\
&x_1(0) = 0, x_1(t_f) = L \\
&x_2(0) = 0, x_2(t_f) = 0
\end{aligned}
$$

$$\text{Hamiltonian}: \; H = \lambda_1 x_2 + \lambda_2 u + \lambda_3$$

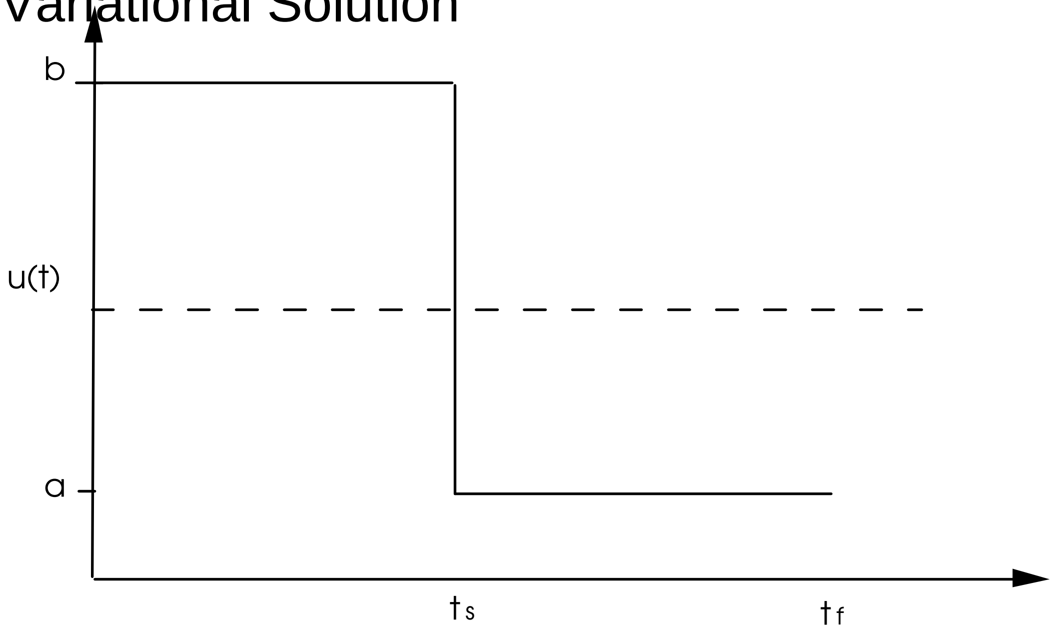
$$
\begin{aligned}
\text{Adjoints}: \; &\dot{\lambda}_1 = 0 ==> \lambda_1(t) = c_1 \\
&\dot{\lambda}_2 = -\lambda_1 ==> \lambda_2(t) = c_2 + c_1(t_f - t) \\
&\dot{\lambda}_3 = 0 ==> \lambda_3(t_f) = 1, \;\; \lambda_3(t) = 1
\end{aligned}
$$

$$\frac{\partial H}{\partial u} = \lambda_2 = c_2 + c_1(t_f - t)\begin{cases} t = 0, c_1 t_f + c_2 > 0, u = b \\ t = t_f, c_2 > 0, u = a \end{cases}$$

Crossover ($\lambda_2 = 0$) occurs at $t = t_s$

Car Problem

Analytic Variational Solution



Optimal Profile

From state equations:

$$x_1(t) = \begin{cases} 1/2\ bt^2, t < t_s \\ 1/2\left(bt_s^2 - a(t_s - t_f)^2\right), t \ge t_s \end{cases}$$

$$x_2(t) = \begin{cases} bt, \quad t < t_s \\ bt_s + a(t - t_s), \quad t \ge t_s \end{cases}$$

•Problem is linear in u(t). Frequently these problems have "bang-bang" character.
•For nonlinear and larger problems, the variational conditions can be solved numerically as boundary value problems.

Apply boundary conditions at $t = t_f$:

$x_1(t_f) = 1/2\ (b\ t_s^2 - a\ (t_s - t_f)^2) = L$

$x_2(t_f) = bt_s + a\ (t_f - t_s) = 0$

$$\Rightarrow\ ts = \left[\frac{2L}{b(1 - b/a)}\right]^{1/2}$$

$$t_f = (1 - b/a)\left[\frac{2L}{b(1 - b/a)}\right]^{1/2}$$

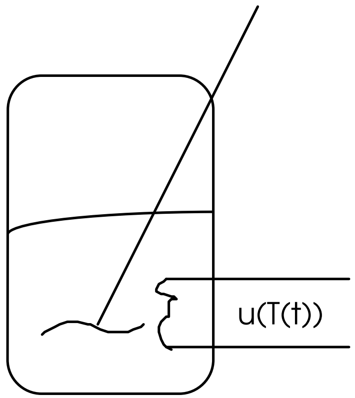
Example: Batch reactor - temperature profile

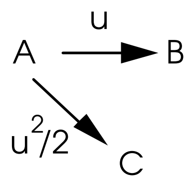
Maximize yield of B after one hour's operation by manipulating a transformed temperature, u(t).

$\Rightarrow$ Minimize $-z_B(1.0)$

s.t.

$$z'_A = -(u+u^2/2)\, z_A$$
$$z'_B = u\, z_A$$
$$z_A(0) = 1$$
$$z_B(0) = 0$$
$$0 \le u(t) \le 5$$





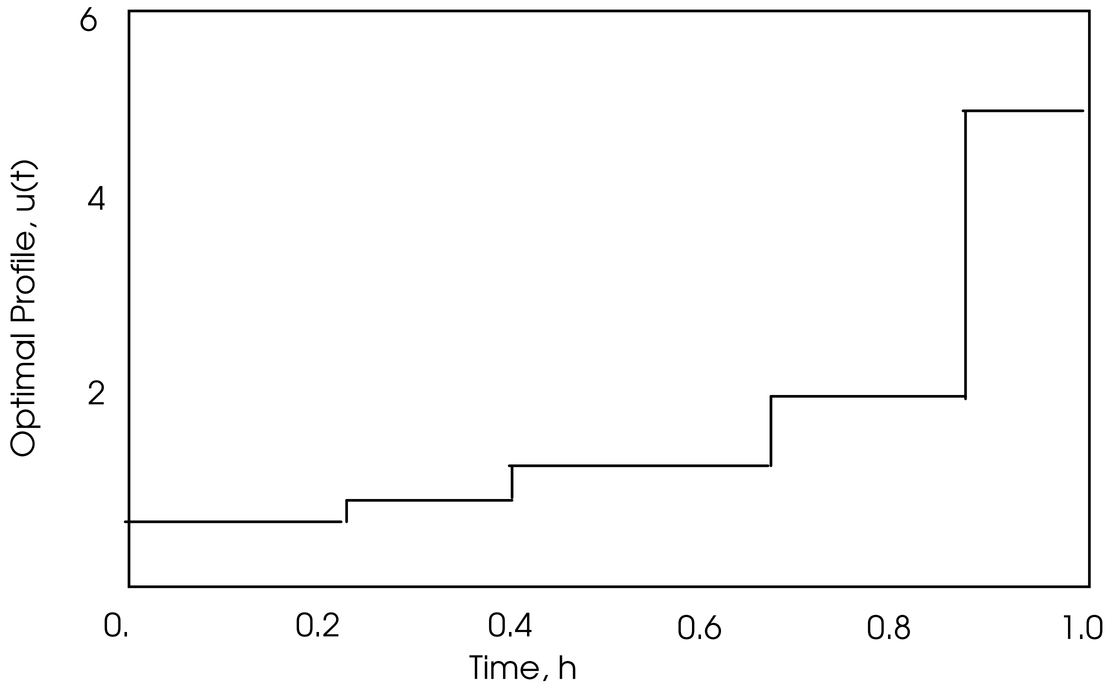
Adjoint Equations:

$$H = -\lambda_A(u+u^2/2)\, z_A + \lambda_B\, u\, z_A$$
$$\partial H/\partial u = \lambda_A\,(1+u)\, z_A + \lambda_B\, z_A$$
$$\lambda'_A = \lambda_A(u+u^2/2) - \lambda_B\, u, \quad \lambda_A(1.0) = 0$$
$$\lambda'_B = 0, \qquad \lambda_B(1.0) = -1$$

Cases Considered

1. NLP Approach - piecewise constant and linear profiles.
2. Control Vector Iteration

Batch Reactor Optimal Temperature Program
Piecewise Constant

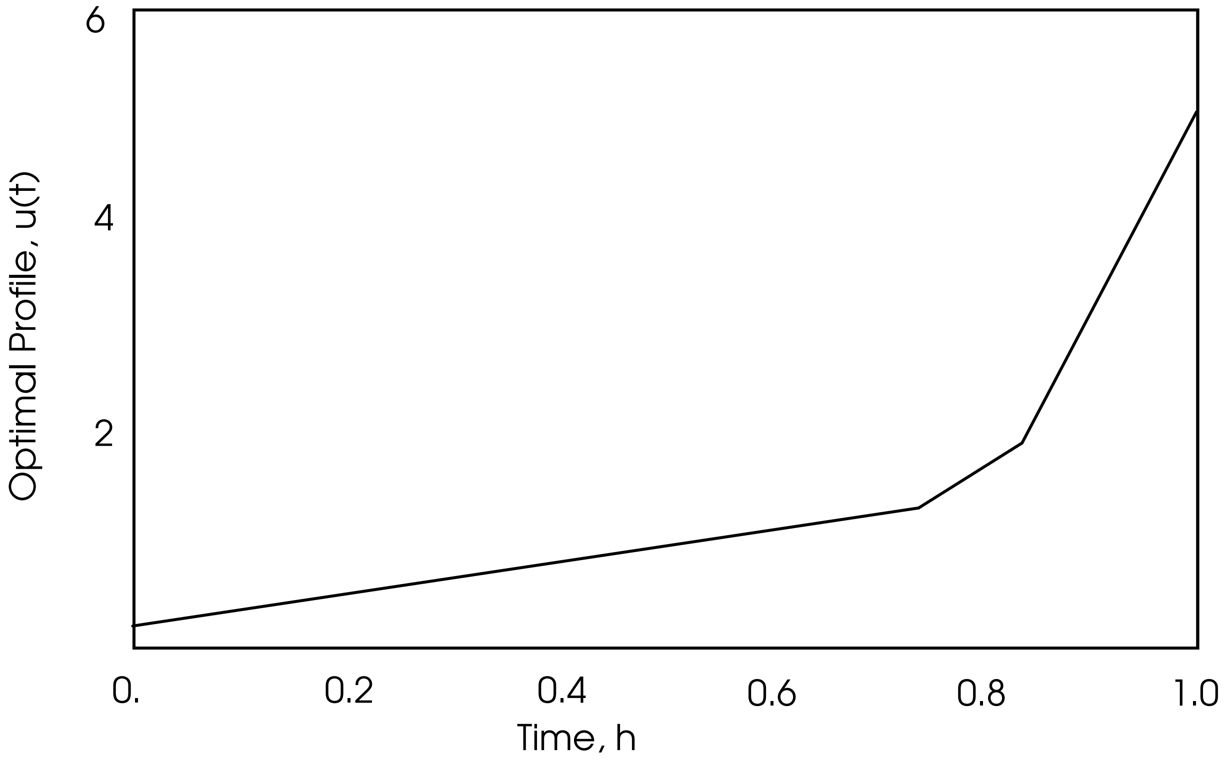


<u>Results</u>

Piecewise Constant Approximation with Variable Time Elements

Optimum B/A: <u>0.57105</u>

Batch Reactor Optimal Temperature Program
Piecewise Linear



Results:

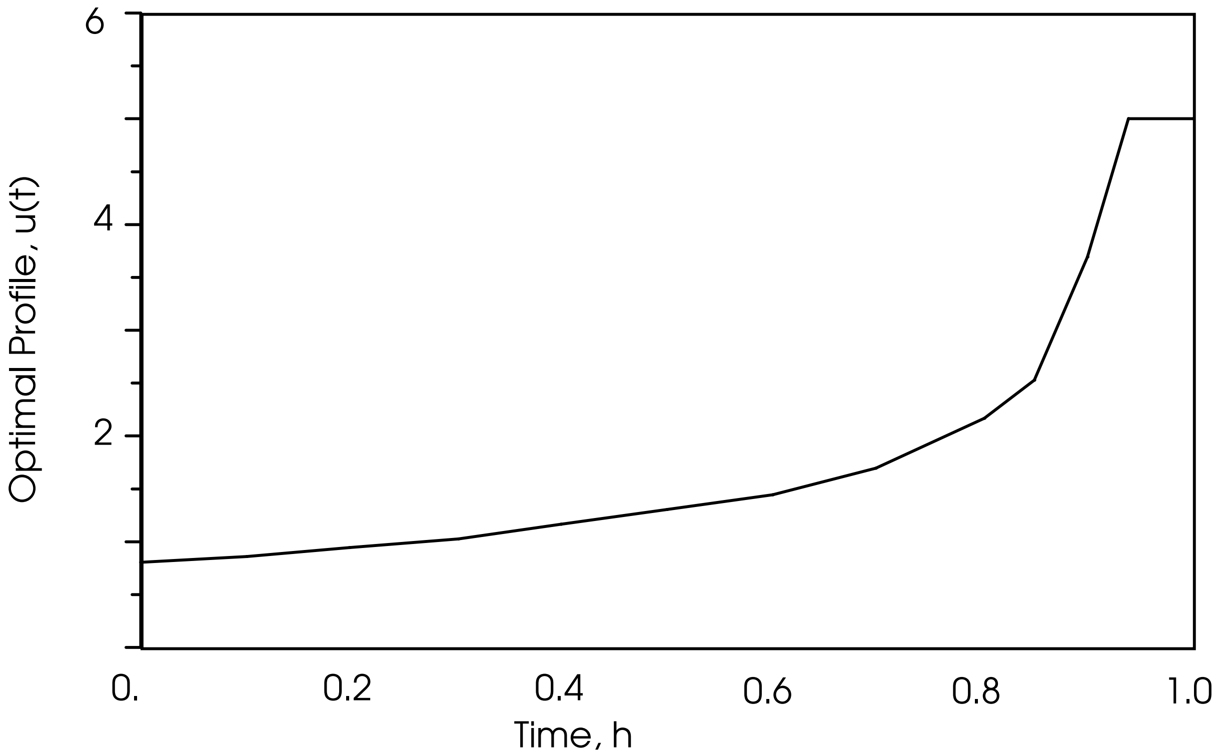
Piecewise Linear Approximation with Variable Time Elements

Optimum B/A: 0.5726

Equivalent # of ODE solutions: 32

Batch Reactor Optimal Temperature Program

Indirect Approach



<u>Results:</u>

Control Vector Iteration with Conjugate Gradients

Optimum (B/A): <u>0.5732</u>

Equivalent # of ODE solutions: <u>58</u>

Dynamic Optimization - Sequential Strategies

Small NLP problem, O(np+nu) (large-scale NLP solver not required)

- Use NPSOL, NLPQL, etc.
- Second derivatives difficult to get

Repeated solution of DAE model and sensitivity/adjoint equations, scales with nz and np

- Dominant computational cost
- May fail at intermediate points

Sequential optimization is not recommended for unstable systems. State variables blow up at intermediate iterations for control variables and parameters.

Discretize control profiles to parameters (at what level?)

Path constraints are difficult to handle exactly for NLP approach

Instabilities in DAE Models

This example cannot be solved with sequential methods (Bock, 1983):

$$dy_1/dt = y_2$$

$$dy_2/dt = \tau^2\, y_1 + (\pi^2 - \tau^2) \sin(\pi\, t)$$

The characteristic solution to these equations is given by:

$$y_1(t) = \sin(\pi\, t) + c_1 \exp(-\tau\, t) + c_2 \exp(\tau\, t)$$

$$y_2(t) = \pi \cos(\pi\, t) - c_1\, \tau \exp(-\tau\, t) + c_2\, \tau \exp(\tau\, t)$$

Both c_1 and c_2 can be set to zero by either of the following equivalent conditions:

IVP $\quad y_1(0) = 0,\ y_2(0) = \pi$

BVP $\quad y_1(0) = 0,\ y_1(1) = 0$

IVP Solution

If we now add roundoff errors e_1 and e_2 to the IVP and BVP conditions, we see significant differences in the sensitivities of the solutions.

For the IVP case, the sensitivity to the *analytic* solution profile is seen by large changes in the profiles $y_1(t)$ and $y_2(t)$ given by:

$$y_1(t) = \sin(\pi t) + (e_1 - e_2/\tau)\exp(-\tau t)/2$$
$$+(e_1 + e_2/\tau)\exp(\tau t)/2$$

$$y_2(t) = \pi\cos(\pi t) - (\tau e_1 - e_2)\exp(-\tau t)/2$$
$$+(\tau e_1 + e_2)\exp(\tau t)/2$$

Therefore, even if e_1 and e_2 are at the level of machine precision ($< 10^{-13}$), a large value of τ and t will lead to unbounded solution profiles.

BVP Solution

On the other hand, for the boundary value problem, the errors affect the *analytic* solution profiles in the following way:

$$y_1(t) = \sin(\pi t) + [e_1 \exp(\tau) - e_2] \exp(-\tau t)/[\exp(\tau) - \exp(-\tau)]$$

$$+ [e_1 \exp(-\tau) - e_2] \exp(\tau t)/[\exp(\tau) - \exp(-\tau)]$$

$$y_2(t) = \pi \cos(\pi t) - \tau [e_1 \exp(\tau) - e_2] \exp(-\tau t)/[\exp(\tau) - \exp(-\tau)]$$

$$+ \tau [e_1 \exp(-\tau) - e_2] \exp(\tau t)/[\exp(\tau) - \exp(-\tau)]$$

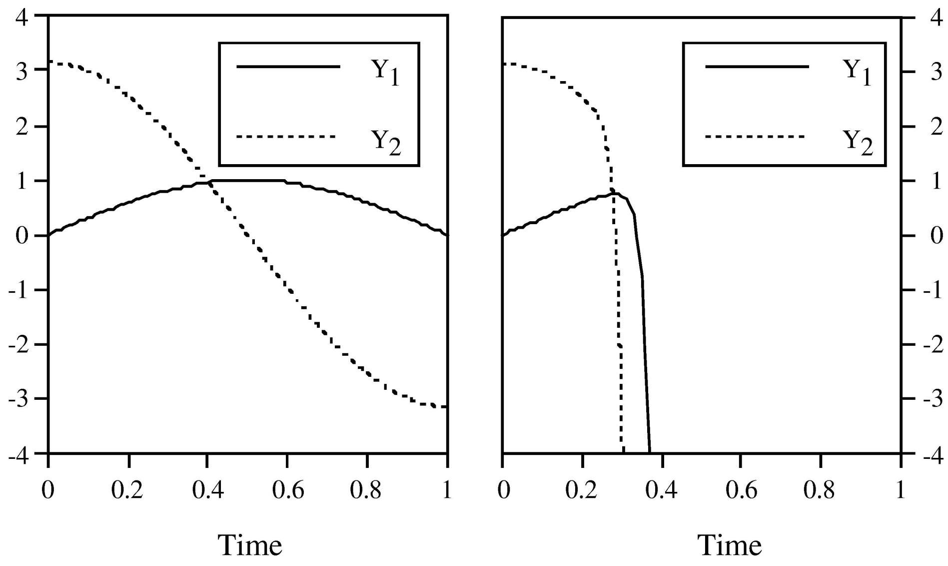
Errors in these profiles never exceed t ($e_1 + e_2$), and as a result a solution to the BVP is readily obtained.

BVP and IVP Profiles

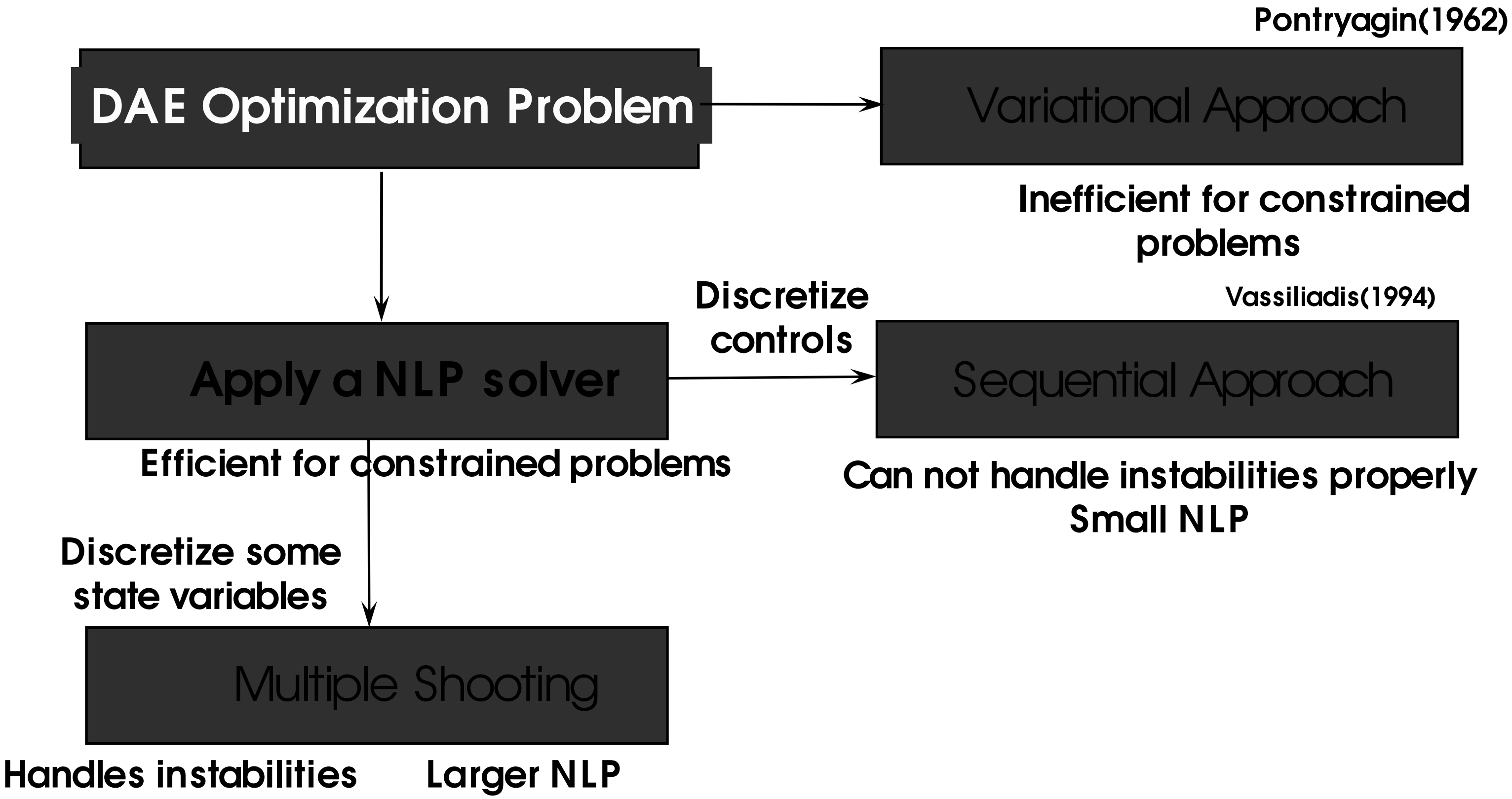
$e_1, e_2 = 10^{-9}$

Linear BVP solves easily

IVP blows up before midpoint

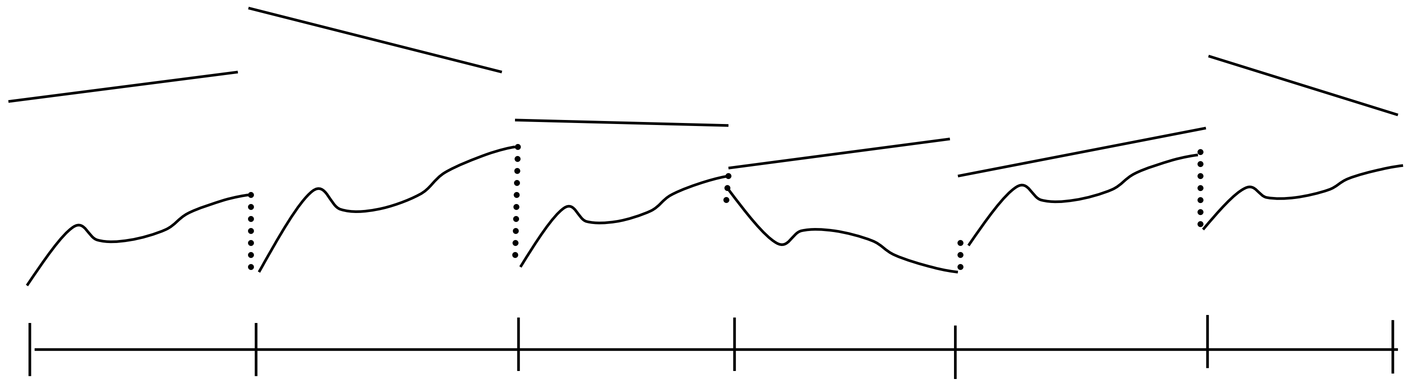


Dynamic Optimization Approaches



Multiple Shooting for Dynamic Optimization

Divide time domain into separate regions



Integrate DAEs state equations over each region

Evaluate sensitivities in each region as in sequential approach wrt u_{ij}, p and z_j

Impose matching constraints in NLP for state variables over each region

Variables in NLP are due to control profiles as well as initial conditions in each region

Multiple Shooting
Nonlinear Programming Problem

$$\min_{u_{i,j},p} \psi\left(z(t_f), y(t_f)\right)$$

s.t.

$$z(z_j, u_{i,j}, p, t_{j+1}) - z_{j+1} = 0$$

$$z_k^{\,l} \le z(z_j, u_{i,j}, p, t_k) \le z_k^{\,u}$$

$$y_k^{\,l} \le y(z_j, u_{i,j}, p, t_k) \le y_k^{\,u}$$

$$u_i^{\,l} \le u_{i,j} \le u_i^{\,u}$$

$$p^l \le p \le p^u$$

$$\left(\frac{dz}{dt}\right) = f\left(z, y, u_{i,j}, p\right), \quad z(t_j) = z_j$$

$$g\left(z, y, u_{\mathrm{i,j}}, p\right) = 0$$

$$z_0^{\,o} = z(0)$$

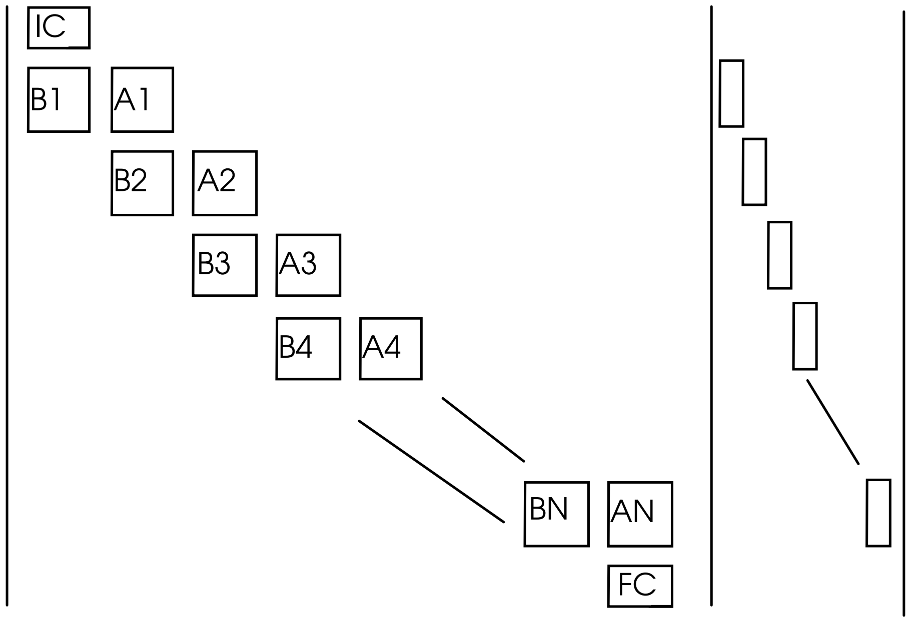
$$\min_{x \in \Re^n} f(x)$$

$$\text{s.t} \quad c(x) = 0$$

$$x^L \le x \le x^u$$

Solved Implicitly

BVP Problem Decomposition



Consider: Jacobian of Constraint Matrix for NLP

• bound unstable modes with boundary conditions (dichotomy)

• can be done implicitly by determining stable pivot sequences in multiple shooting constraints approach

• well-conditioned problem implies dichotomy in BVP problem (deHoog and Mattheij)

Bock Problem (with t = 50)

• Sequential approach blows up (starting within 10^{-9} of optimum)

• Multiple Shooting optimization requires 4 SQP iterations

Dynamic Optimization – Multiple Shooting Strategies

Larger NLP problem O(np+nu+NE nz)

- Use SNOPT, MINOS, etc.
- Second derivatives difficult to get

Repeated solution of DAE model and sensitivity/adjoint equations, scales with nz and np

- Dominant computational cost
- May fail at intermediate points

Multiple shooting can deal with unstable systems with sufficient time elements.

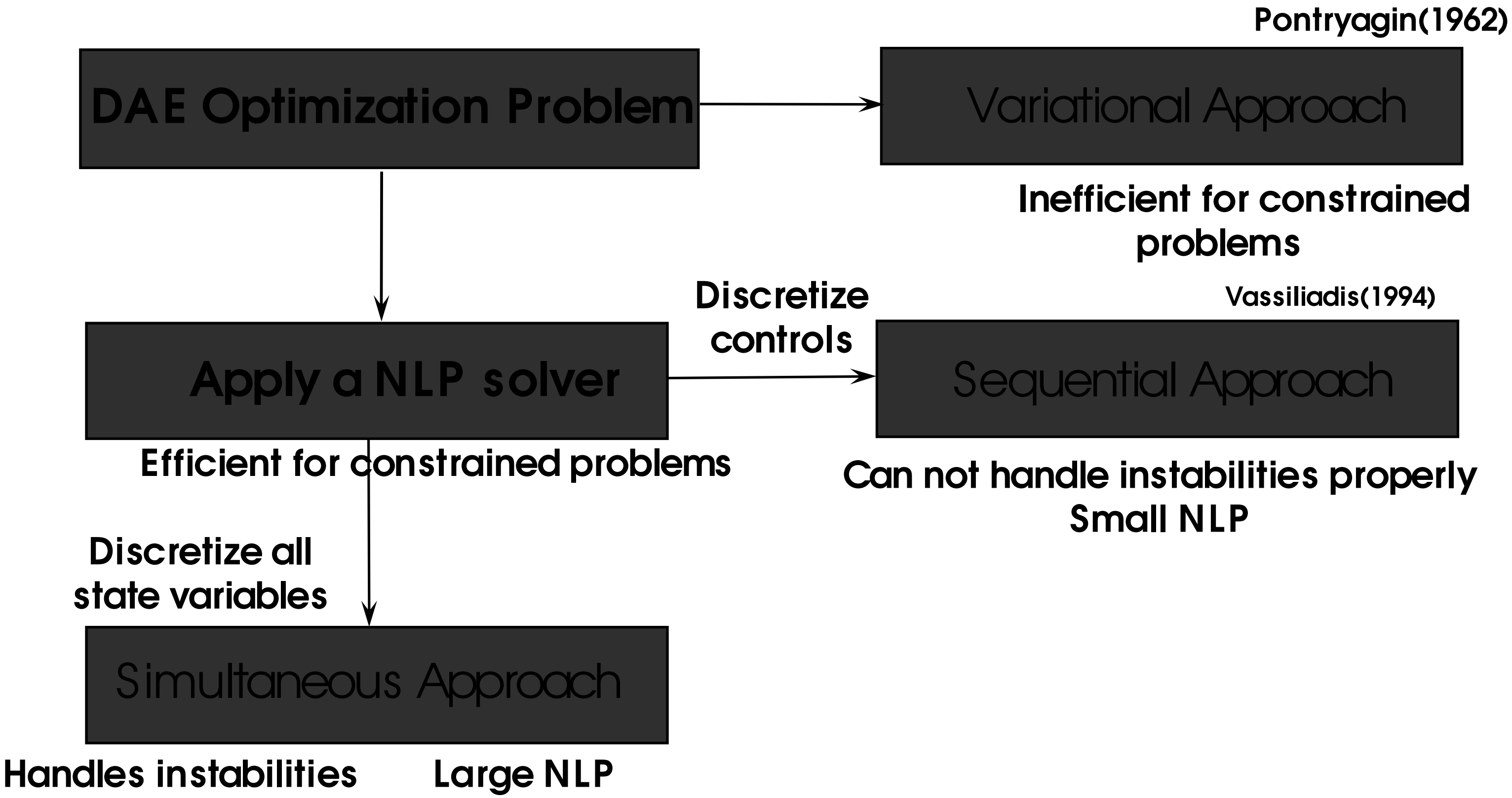
Discretize control profiles to parameters (at what level?)

Path constraints are difficult to handle exactly for NLP approach

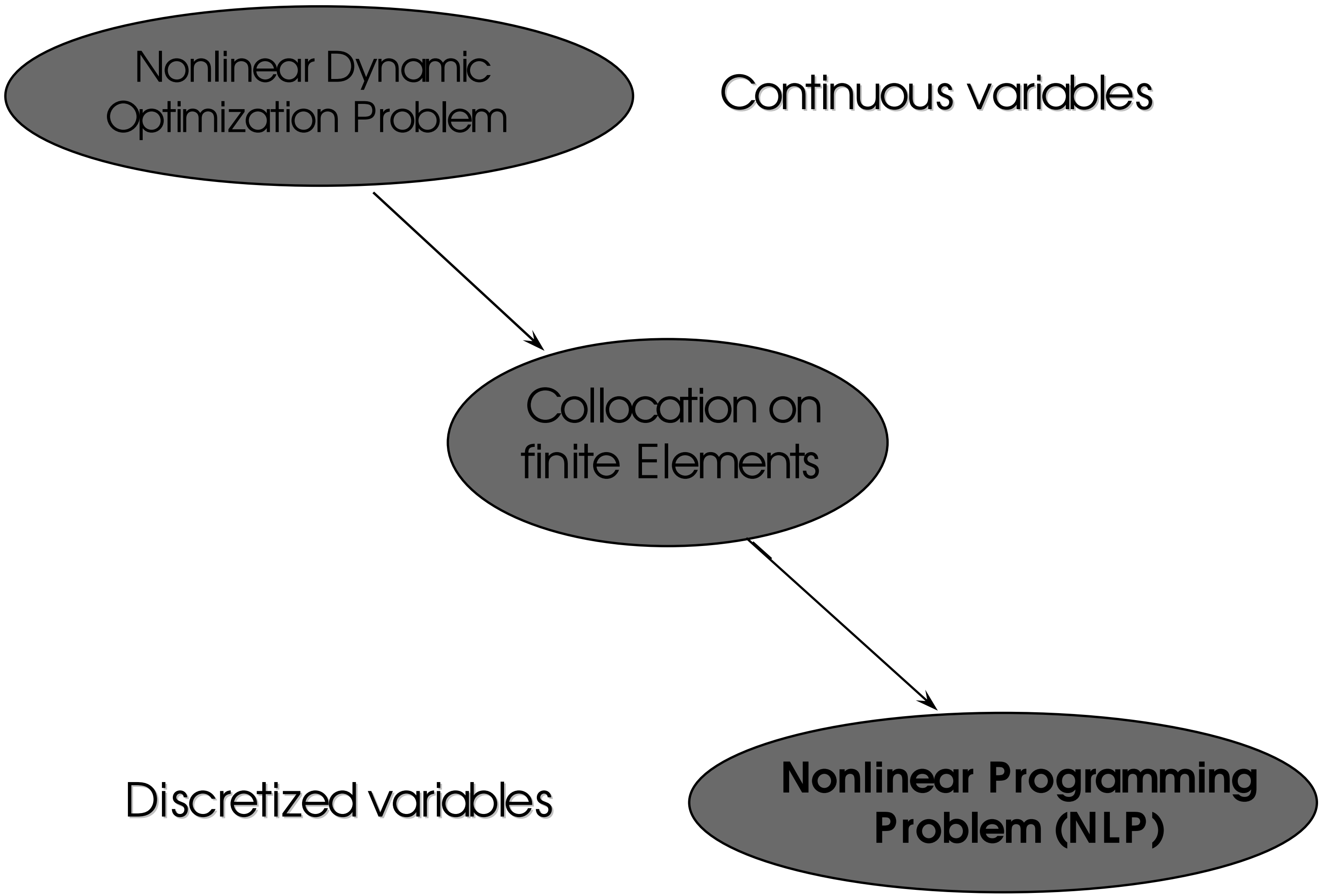
Block elements for each element are dense!

Extensive developments and applications by Bock and coworkers using MUSCOD code

Dynamic Optimization Approaches



Nonlinear Programming Formulation



Discretization of Differential Equations
Orthogonal Collocation

Given: *dz/dt = f(z, u, p), z(0)=given*

Approximate *z* and *u* by Lagrange interpolation polynomials (order K+1 and K, respectively) with interpolation points, t_k

$$z_{K+1}(t) = \sum_{k=0}^{K} z_k \ell_k(t), \ell_k(t) = \prod_{\substack{j=0 \\ j \neq k}}^{K} \frac{(t - t_j)}{(t_k - t_j)} ==> z_{N+1}(t_k) = z_k$$

$$u_K(t) = \sum_{k=1}^{K} u_k \ell_k(t), \ell_k(t) = \prod_{\substack{j=1 \\ j \neq k}}^{K} \frac{(t - t_j)}{(t_k - t_j)} ==> u_N(t_k) = u_k$$

Substitute z_{N+1} and u_N into ODE and apply equations at t_k.

$$r(t_k) = \sum_{j=0}^{K} z_j \dot{\ell}_j(t_k) - f(z_k, u_k) = 0, \quad k = 1,...K$$

Collocation Example

$$z_{K+1}(t) = \sum_{k=0}^{K} z_k \ell_k(t), \ell_k(t) = \prod_{\substack{j=0 \\ j\neq k}}^{K} \frac{(t-t_j)}{(t_k - t_j)} ==> z_{N+1}(t_k) = z_k$$

$t_0 = 0, t_1 = 0.21132, t_2 = 0.78868$

$\ell_0(t) = (t^2 - t + 1)/6, \quad \dot{\ell}_0(t) = t/3 - 1/6$

$\ell_1(t) = -8.195\, t2 + 6.4483\, t, \quad \dot{\ell}_1(t) = 6.4483 - 16.39\, t$

$\ell_2(t) = 2.19625\, t2 - 0.4641\, t, \quad \dot{\ell}_2(t) = 4.392\, t - 0.46412$

Solve $z' = z^2 - 3z + 2, z(0) = 0$

$==> z_0 = 0$

$z_0 \dot{\ell}_0(t_1) + z_1 \dot{\ell}_1(t_1) + z_2 \dot{\ell}_2(t_1) = z_1^2 - 3z_1 + 2$

$(2.9857\, z_1 + 0.46412\, z_2 = z_1^2 - 3z_1 + 2)$

$z_0 \dot{\ell}_0(t_2) + z_1 \dot{\ell}_1(t_2) + z_2 \dot{\ell}_2(t_2) = z_2^2 - 3z_2 + 2$

$(-6.478\, z_1 + 3z_2 = z_2^2 - 3z_2 + 2)$

$z_0 = 0, z_1 = 0.291\ (0.319), z_2 = 0.7384\ (0.706)$

$z(t) = 1.5337\, t - 0.76303\, t^2$

Converted Optimal Control Problem Using Collocation

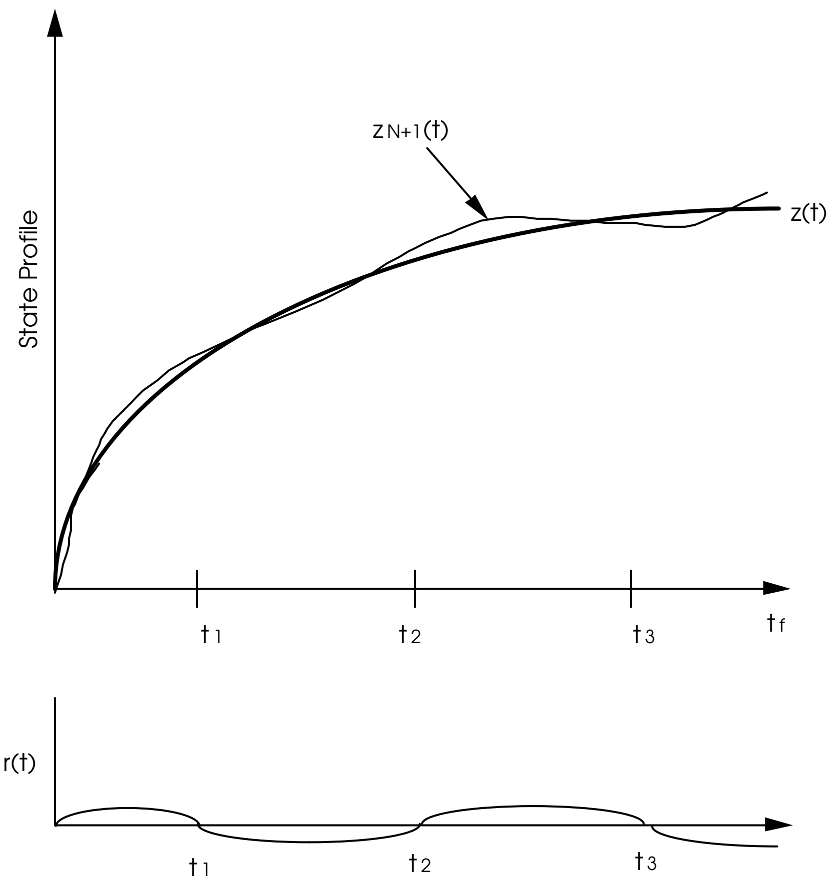
$$\begin{aligned} Min \quad & \phi(z(t_f)) \\ s.t. \quad & z' = f(z, u, p),\ z(0)=z_0 \\ & g(z(t), u(t), p) \le 0 \\ & h(z(t), u(t), p) = 0 \end{aligned}$$

to Nonlinear Program

$$Min \quad \phi(z_f)$$

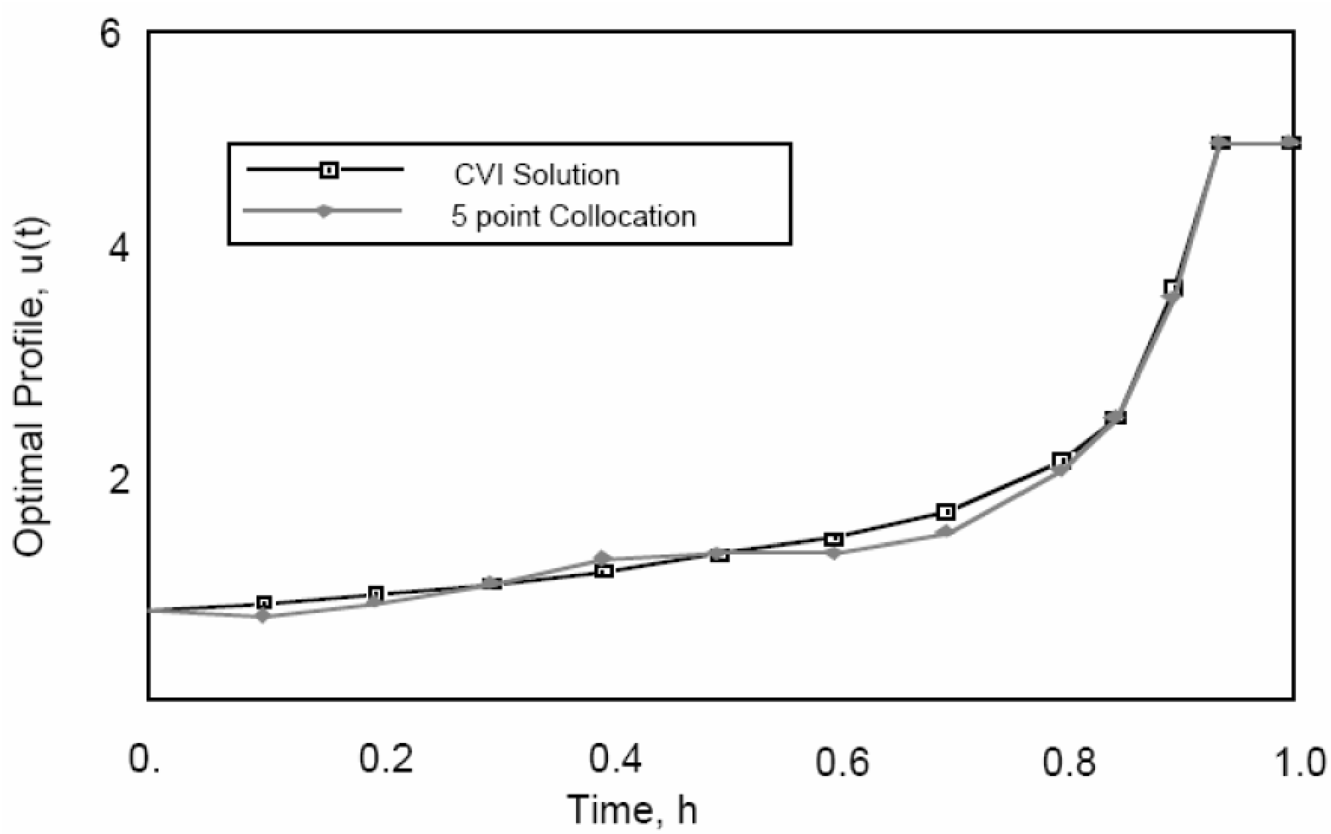
$$\left.\begin{aligned} & \sum_{j=0}^{K} z_j \dot{\ell}_j(t_k) - f(z_k, u_k) = 0,\ z_0 = z(0) \\ & g(z_k, u_k) \le 0 \\ & h(z_k, u_k) = 0 \end{aligned}\right\} k = 1, \ldots K$$

$$\sum_{j=0}^{K} z_j \ell_j(1) - z_f = 0$$



How accurate is approximation

Results of Optimal Temperature Program
Batch Reactor (Revisited)



Results - NLP with Orthogonal Collocation

Optimum B/A - 0.5728

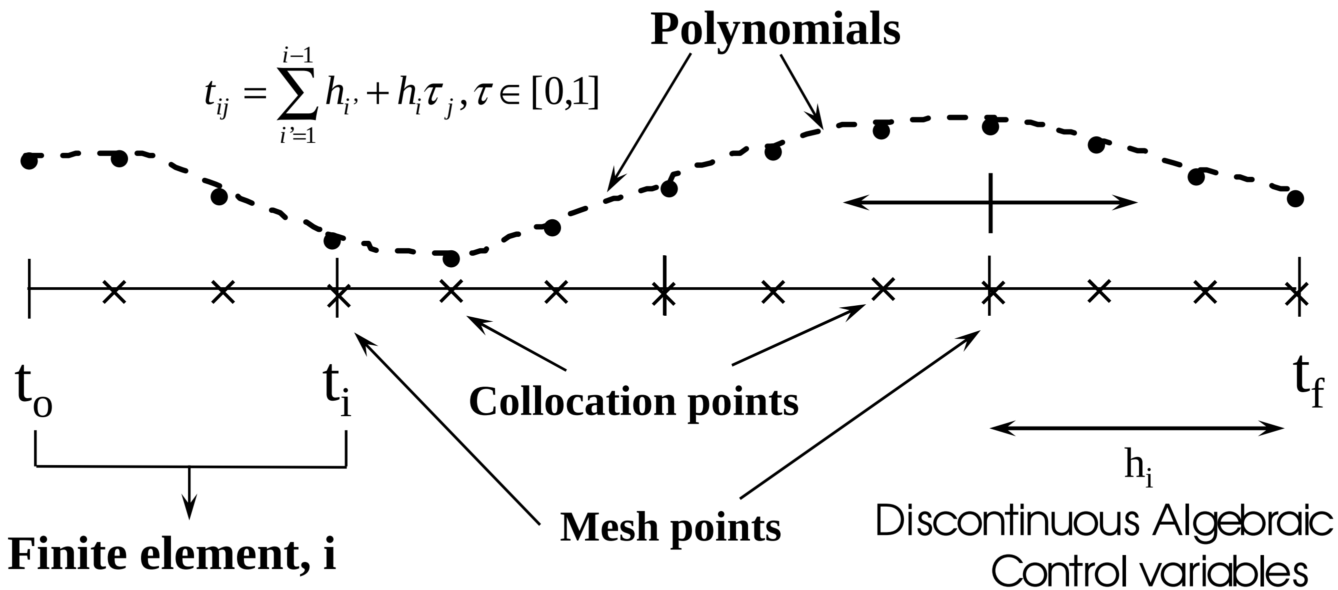
# of ODE Solutions - 0.7 (Equivalent)

Collocation on Finite Elements

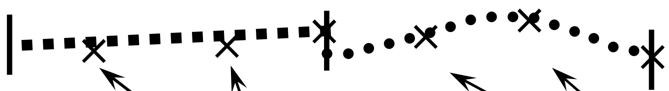
$$t_{ij} = \sum_{i'=1}^{i-1} h_{i'} + h_i \tau_j, \; \tau \in [0,1]$$

$$\frac{dz}{dt} = \frac{1}{h_i}\frac{dz}{d\tau}$$

$$\frac{dz}{d\tau} = h_i f(z,u)$$

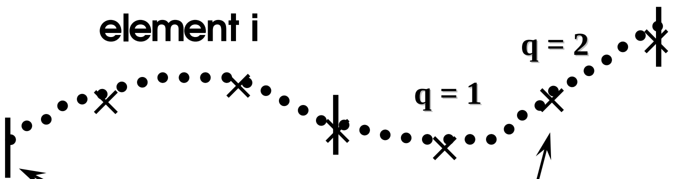


Discontinuous Algebraic and Control variables



$$y(t) = \sum_{q=1}^{K} \bar{\ell}_q(t)\, y_{iq} \qquad u(t) = \sum_{q=1}^{K} \bar{\ell}_q(t)\, u_{iq}$$

Continuous Differential variables



$$z(t) = \sum_{q=0}^{K} \ell_q(t)\, z_{iq}$$

$$r(t_{ik}) = \sum_{j=0}^{K} (z_{ij} \dot{\ell}_j(\tau_k)) - h_i f(z_{ik}, u_{ik}, p) = 0, \quad \mathrm{k} = 1,..\mathrm{K}, \mathrm{i} = 1,..\,\mathrm{NE}$$

Nonlinear Programming Problem

$$\min \; \psi\left(z_f\right)$$

s.t.

$$\sum_{j=0}^{K}(z_{ij}\dot{\ell}_j(\tau_k)) - h_i f(z_{ik}, u_{ik}, p) = 0$$

$$g\left(z_{i,k}, y_{i,k}, u_{i,k}, p\right) = 0$$

$$\sum_{j=0}^{K}(z_{i-1,j}\ell_j(1)) - z_{i0} = 0,\; i = 2,..NE$$

$$\sum_{j=0}^{K}(z_{NE,j}\ell_j(1)) - z_f = 0,\; z_{10} = z(0)$$

$$z_{\mathrm{i,j}}^{\,l} \le z_{\mathrm{i,j}} \le z_{\mathrm{i,j}}^{\,u}$$

$$y_{i,j}^{\,l} \le y_{\mathrm{i,j}} \le y_{i,j}^{\,u}$$

$$u_{i,j}^{\,l} \le u_{i,j} \le u_{i,j}^{\,u}$$

$$p^l \le p \le p^u$$

$$\min_{x \in \Re^n} f(x)$$

$$\text{s.t} \quad c(x) = 0$$

$$x^L \le x \le x^u$$

Finite elements, h_i, can also be variable to determine break points for *u(t)*.

Add $h_u \geq h_i \geq 0, \sum h_i = t_f$

Can add constraints $g(h, z, u) \leq \varepsilon$ for approximation error

Hot Spot Reactor Revisited

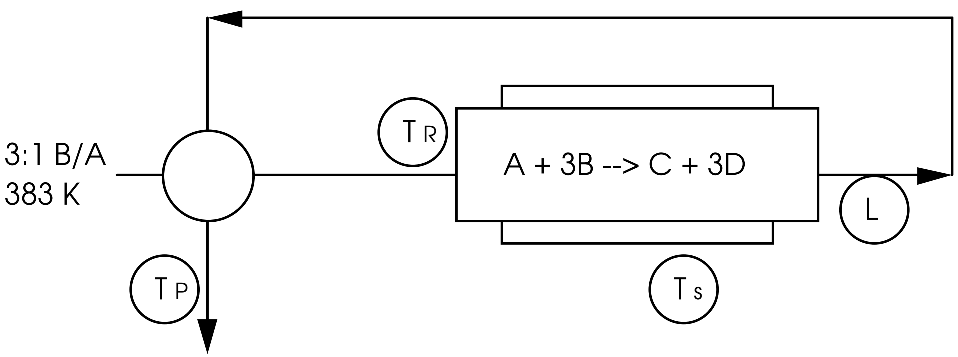
$$\underset{T_P, T_R, L, T_S}{Min} \quad \Phi = L - \int_0^L (T(t) - T_S / T_R)dt$$

$$s.t. \quad \frac{dq}{dt} = 0.3(1 - q(t))\exp[20 - 20/T(t)], \; q(0) = 0$$

$$\frac{dT}{dt} = -1.5(T(t) - T_S / T_R) + 2/3 \frac{dq}{dt}, \; T(0) = 1$$

$$\Delta H_{feed}(T_R, 110^o C) - \Delta H_{product}(T_P, T(L)) = 0$$

$$T_P = 120^o C, \; T(L) = 1 + 10^o C/T_R$$



T_P = specified product temperature
T_R = reactor inlet, reference temperature
L = reactor length
T_S = steam sink temperature
q(t) = reactor conversion profile
T(t) = normalized reactor temperature profile

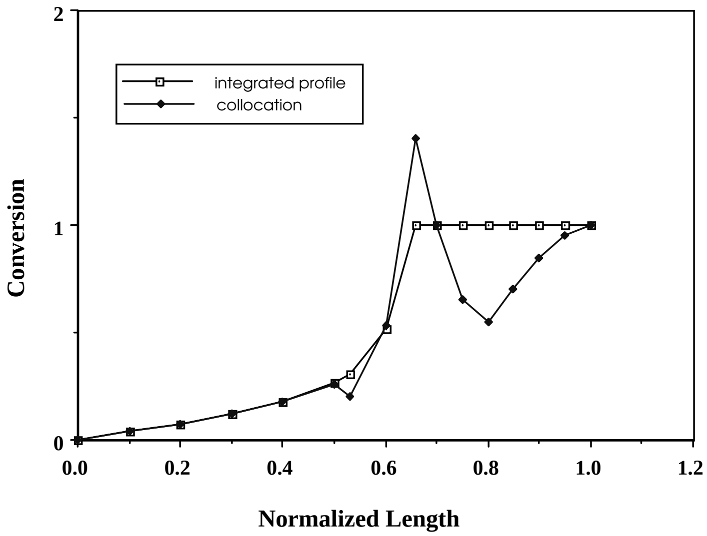
Cases considered:

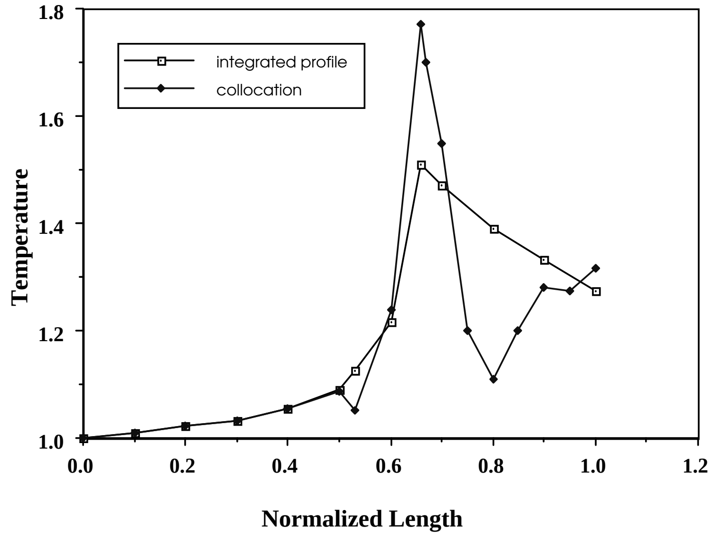
- Hot Spot - no state variable constraints
- Hot Spot with $T(t) \leq 1.45$

Base Case Simulation

Method: OCFE at initial point with 6 equally spaced elements

	L(norm)	T_R(K)	T_S(K)	T_P(K)
Base Case:	1.0	462.23	425.26	250



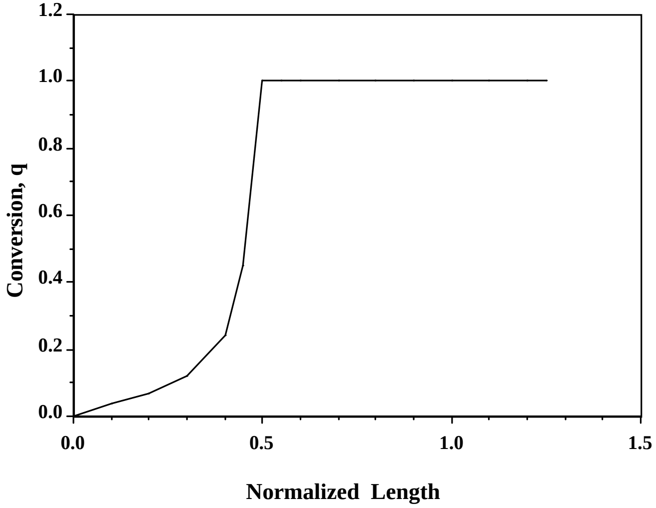


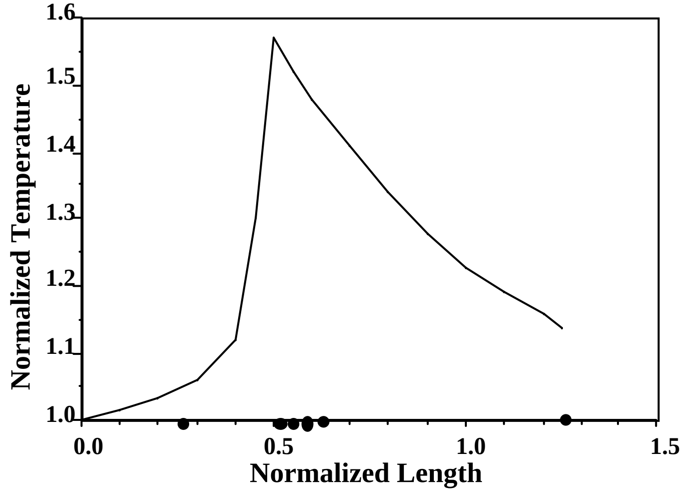
Unconstrained Case

Method: OCFE combined formulation with rSQP
identical to integrated profiles at optimum

	L(norm)	T_R(K)	T_S(K)	T_P(K)
Initial:	1.0	462.23	425.26	250
Optimal:	1.25	500	470.1	188.4

123 CPU s. (μVax II)
$\phi^* = -171.5$





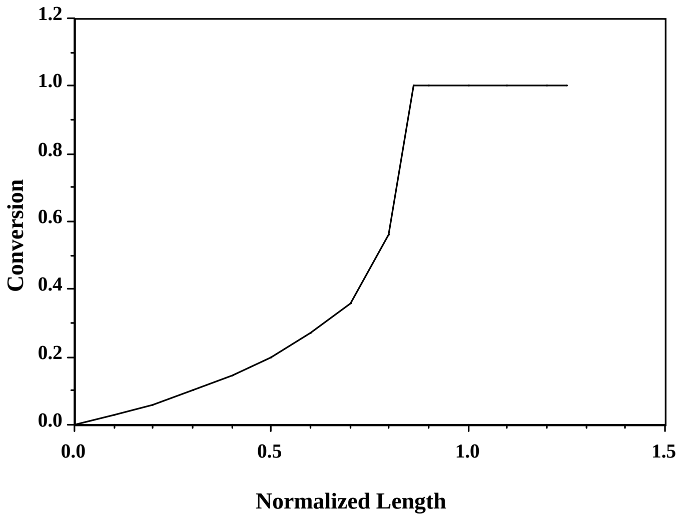
Temperature Constrained Case

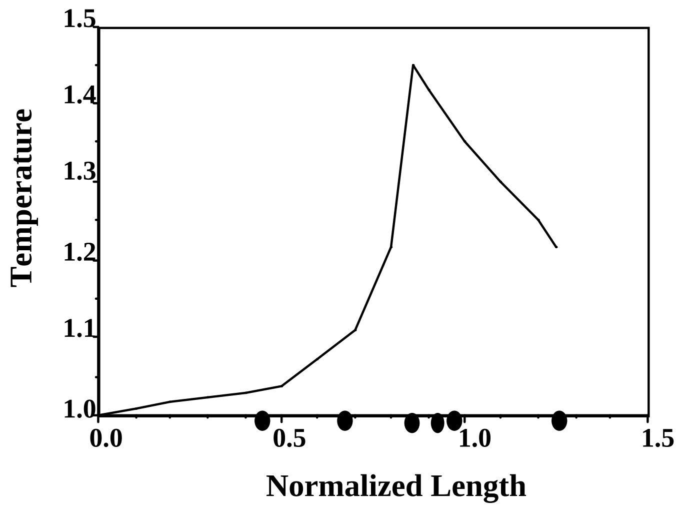
$T(t) \leq 1.45$

Method: OCFE combined formulation with rSQP, identical to integrated profiles at optimum

	L(norm)	T_R(K)	T_S(K)	T_P(K)
Initial:	1.0	462.23	425.26	250
Optimal:	1.25	500	450.5	232.1

57 CPU s. (μVax II), $\phi^* = -148.5$





Theoretical Properties of Simultaneous Method

A. <u>Stability and Accuracy of Orthogonal Collocation</u>

- Equivalent to performing a *fully implicit* Runge-Kutta integration of the DAE models at Gaussian (Radau) points
- 2K order (2K-1) method which uses K collocation points
- Algebraically stable (i.e., possesses A, B, AN and BN stability)

B. <u>Analysis of the Optimality Conditions</u>

- An equivalence has been established between the Kuhn-Tucker conditions of NLP and the variational necessary conditions
- Rates of convergence have been established for the NLP method

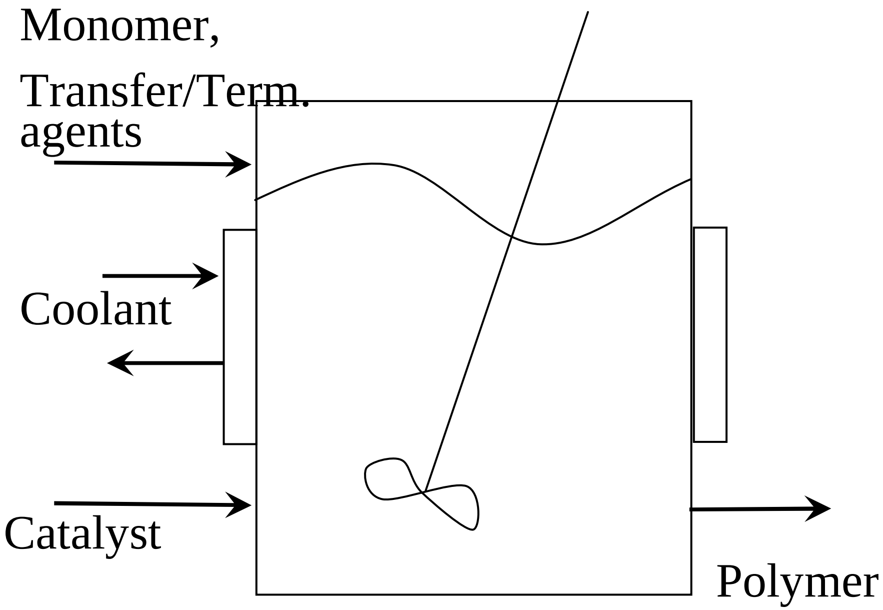
Simultaneous DAE Optimization

Case Studies

- Reactor - Based Flowsheets
- Fed-Batch Penicillin Fermenter
- Temperature Profiles for Batch Reactors
- Parameter Estimation of Batch Data
- Synthesis of Reactor Networks
- Batch Crystallization Temperature Profiles
- Grade Transition for LDPE Process
- Ramping for Continuous Columns
- Reflux Profiles for Batch Distillation and Column Design
- Source Detection for Municipal Water Networks
- Air Traffic Conflict Resolution
- Satellite Trajectories in Astronautics
- Batch Process Integration
- Optimization of Simulated Moving Beds

Production of High Impact Polystyrene (HIPS)

Startup and Transition Policies (Flores et al., 2005a)



Initiation reactions

Thermal

$$3M_S \xrightarrow{k_{i0}} 2R_S^1$$

Chemical

$$\begin{aligned}
I &\xrightarrow{f_1 k_d} 2R \\
R + M_S &\xrightarrow{k_{i1}} R_S^1 \\
R + B_0 &\xrightarrow{k_{i2}} B_R \\
B_R + M_S &\xrightarrow{k_{i3}} B_{RS}^1
\end{aligned}$$

Propagation reactions

$$\begin{aligned}
R_S^j + M_S &\xrightarrow{k_p} R_S^{j+1} \\
B_{RS}^j + M_S &\xrightarrow{k_p} B_{RS}^{j+1}
\end{aligned}$$

Definite termination reactions

Homopolymer

$$R_S^j + R_S^m \xrightarrow{k_t} P^{j+m}$$

Grafting

$$\begin{aligned}
R_S^j + B_R &\xrightarrow{k_t} B_P^j \\
R_S^j + B_{RS}^m &\xrightarrow{k_t} B_P^{j+m}
\end{aligned}$$

Crosslinking

$$\begin{aligned}
B_R + B_R &\xrightarrow{k_t} B_{EB} \\
B_{RS}^j + B_R &\xrightarrow{k_t} B_{PB}^j \\
B_{RS}^j + B_{RS}^m &\xrightarrow{k_t} B_{PB}^{j+m}
\end{aligned}$$

Transfer reactions

Monomer

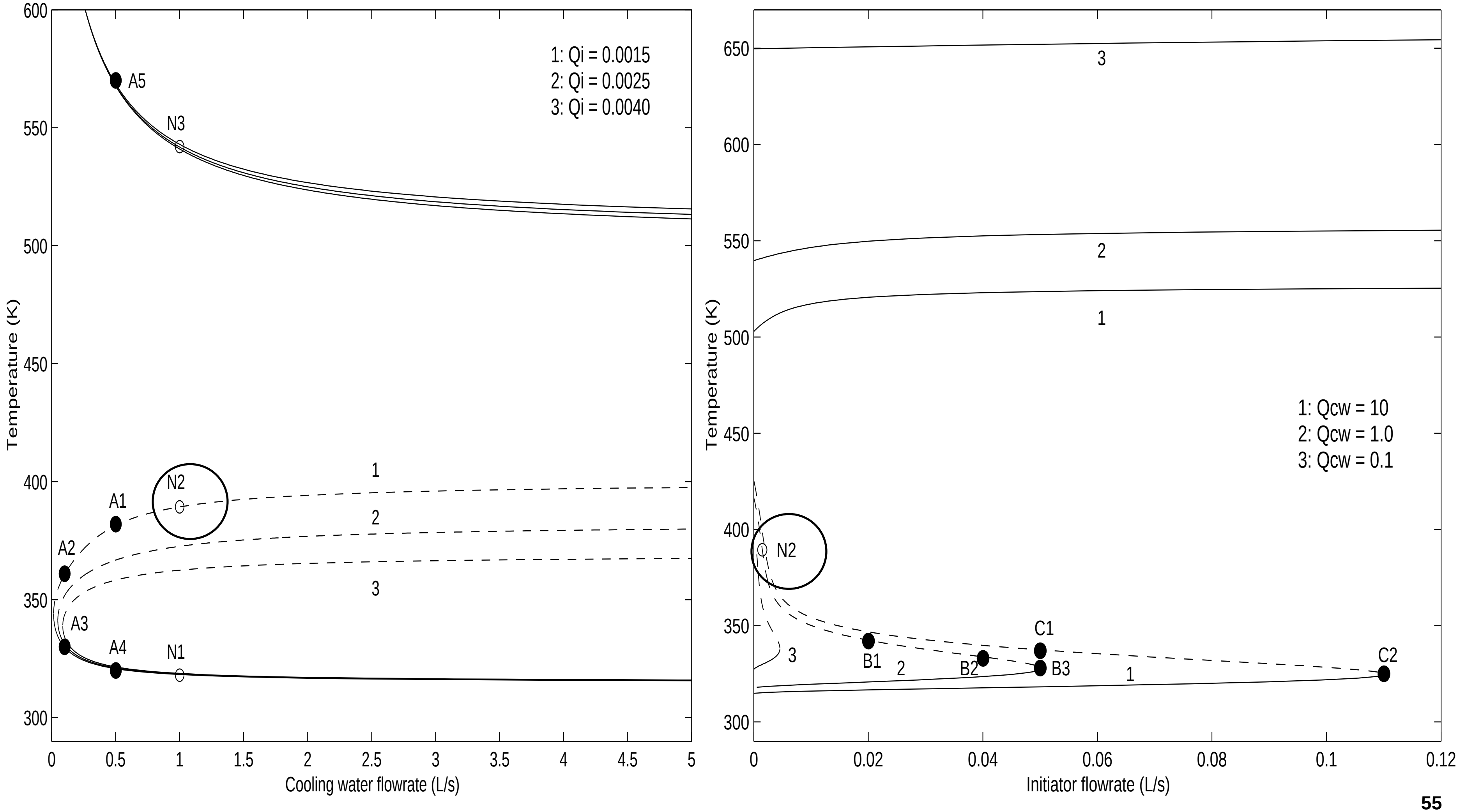
$$\begin{aligned}
R_S^j + M_S &\xrightarrow{k_{fs}} P^j + R_S^1 \\
B_{RS}^j + M_S &\xrightarrow{k_{fs}} B_P^j + R_S^1
\end{aligned}$$

Grafting sites

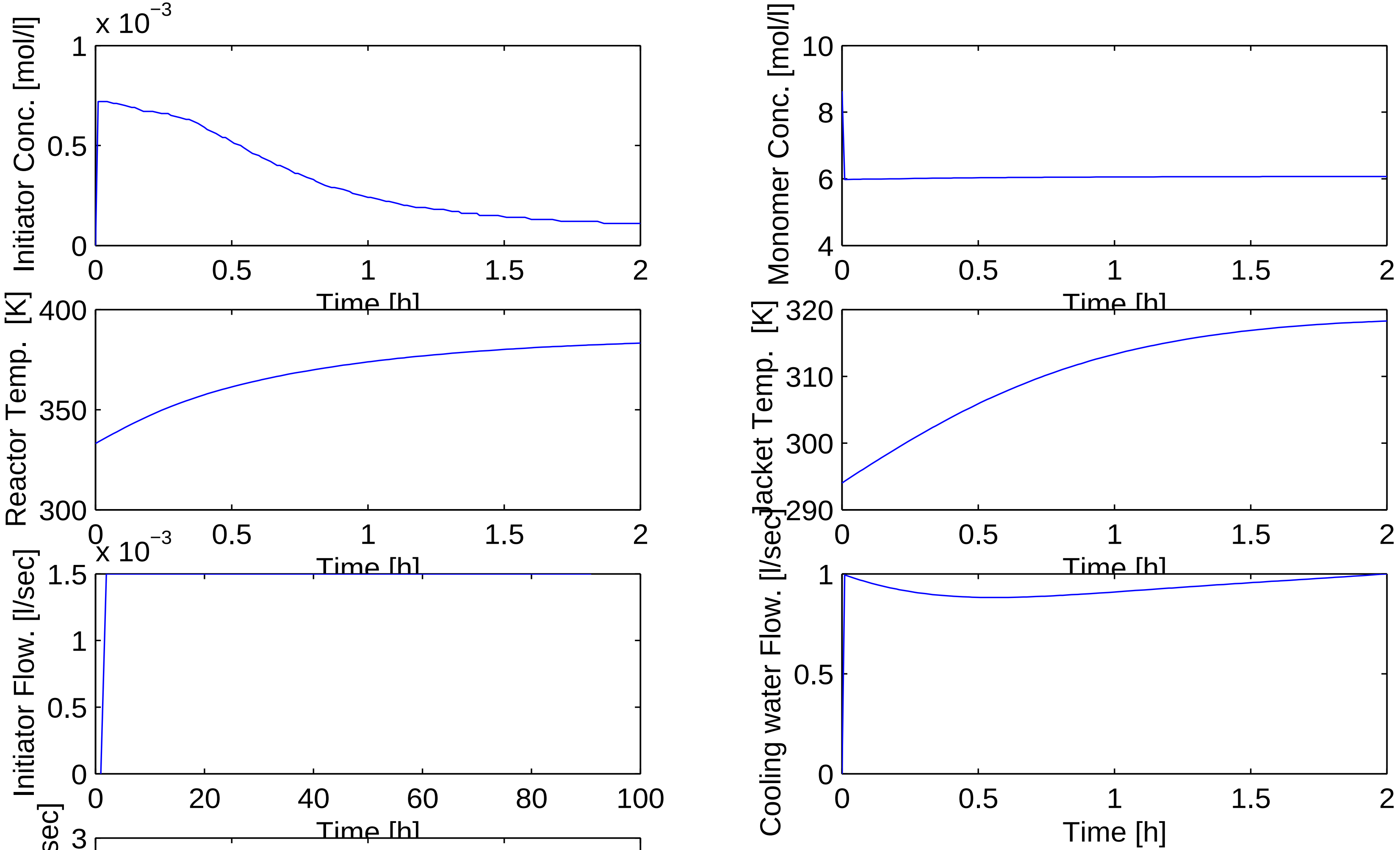
$$\begin{aligned}
R_S^j + B_0 &\xrightarrow{k_{fb}} P^j + B_R \\
B_{RS}^j + B_0 &\xrightarrow{k_{fb}} B_P^j + B_R
\end{aligned}$$

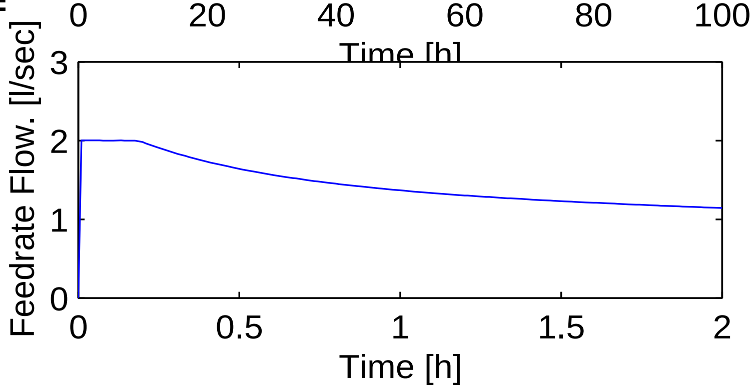
Phase Diagram of Steady States

Transitions considered among all steady state pairs



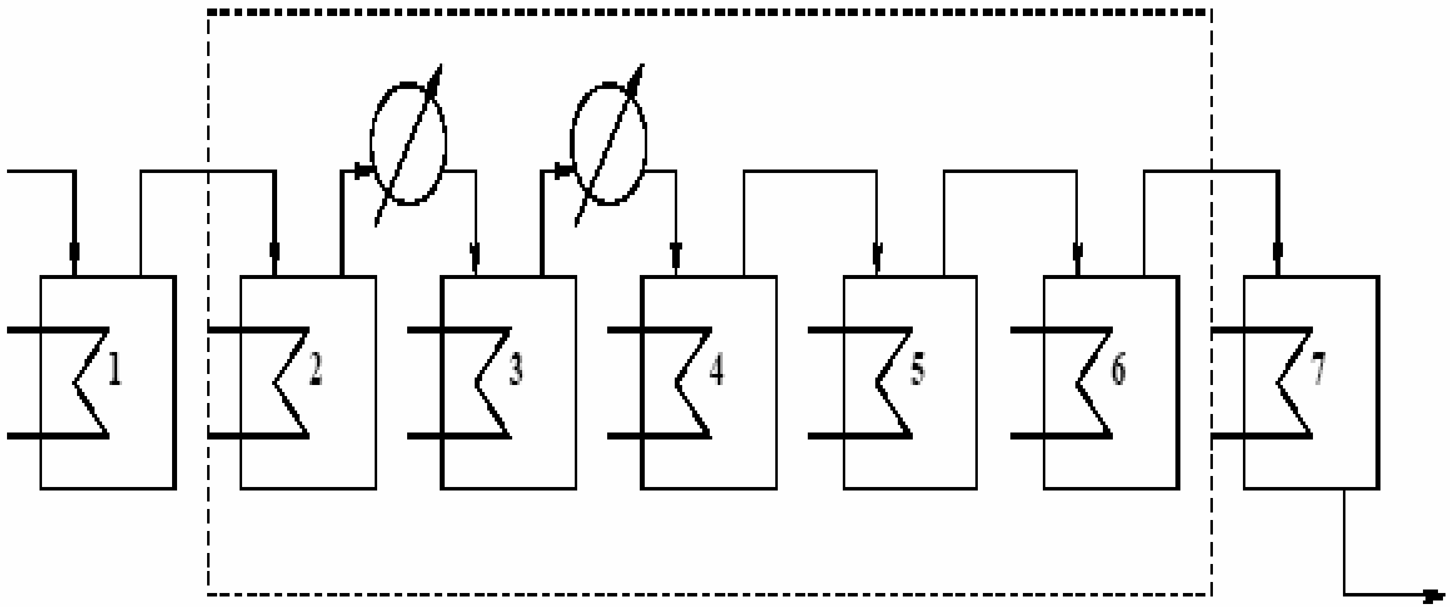
Startup to Unstable Steady State





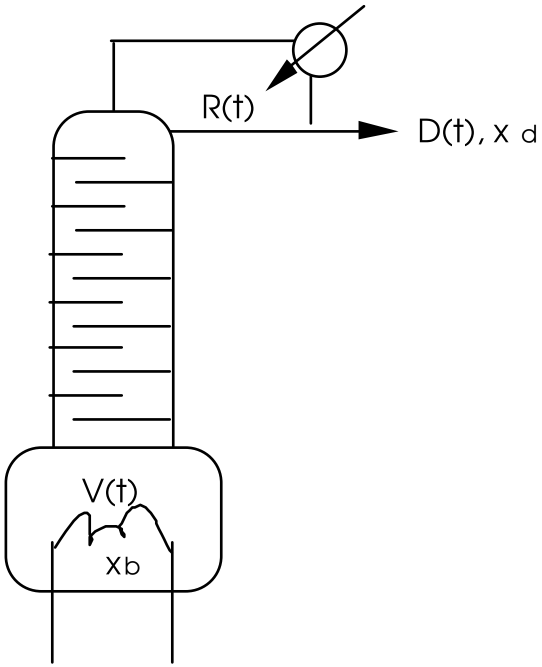
- 926 variables
- 476 constraints
- 36 iters. / 0.95 CPU s (P4)

HIPS Process Plant (Flores et al., 2005b)



- Many grade transitions considered with stable/unstable pairs
- 1-6 CPU min (P4) with IPOPT
- Study shows benefit for sequence of grade changes to achieve wide range of grade transitions.

Batch Distillation – Optimization Case Study - 1



$$\frac{dx_{i,d}}{dt} = \frac{V}{H_{cond}}\left[\left(y_{i,N} - x_{i,d}\right)\right]$$

$$\frac{dx_{i,0}}{dt} = \frac{V}{S}\left[\left(x_{i,d} - x_{i,0}\right)\right]$$

$$\frac{dS}{dt} = \frac{-V}{R+1}$$

- Gauge effect of column holdups
- Overall profit maximization
- Make Tray Count Continuous

Optimization Case Study - 1

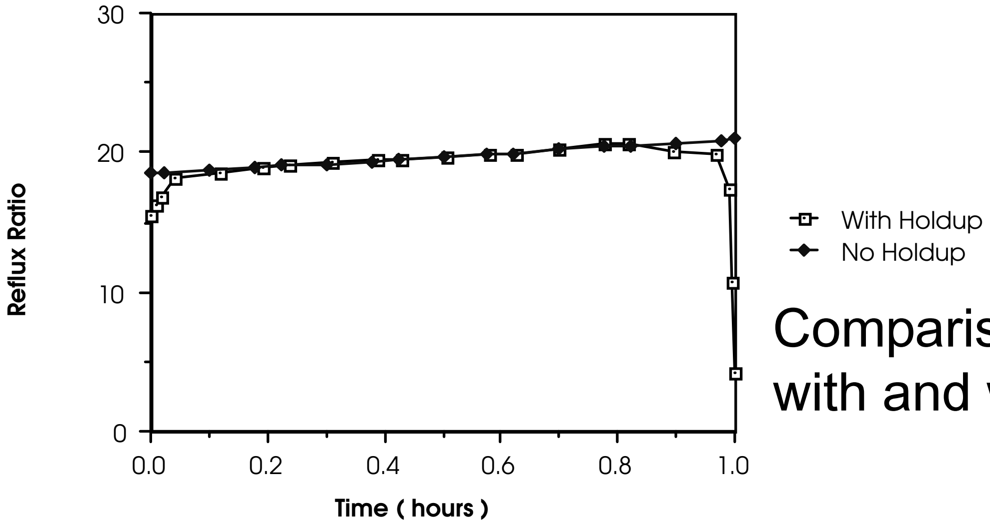
Modeling Assumptions

- Ideal Thermodynamics
- No Liquid Tray Holdup
- No Vapor Holdup
- Four component mixture (α = 2, 1.5, 1.25, 1)
- Shortcut steady state tray model
  (Fenske-Underwood-Gilliland)
- Can be substituted by more detailed steady state models
  (Fredenslund and Galindez, 1988; Alkaya, 1997)

Optimization Problems Considered

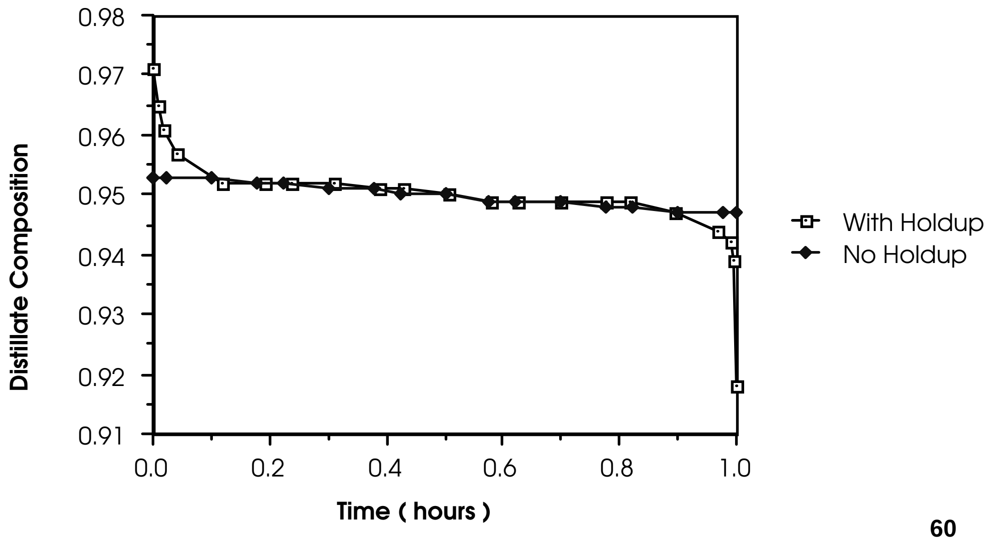
- Effect of Column Holdup (H_{cond})
- Total Profit Maximization

Maximum Distillate Problem



Comparison of optimal reflux profiles with and without holdup (H_{cond})

Comparison of distillate profiles with and without holdup (H_{cond}) at 95.5% overall purity



Batch Distillation Profit Maximization

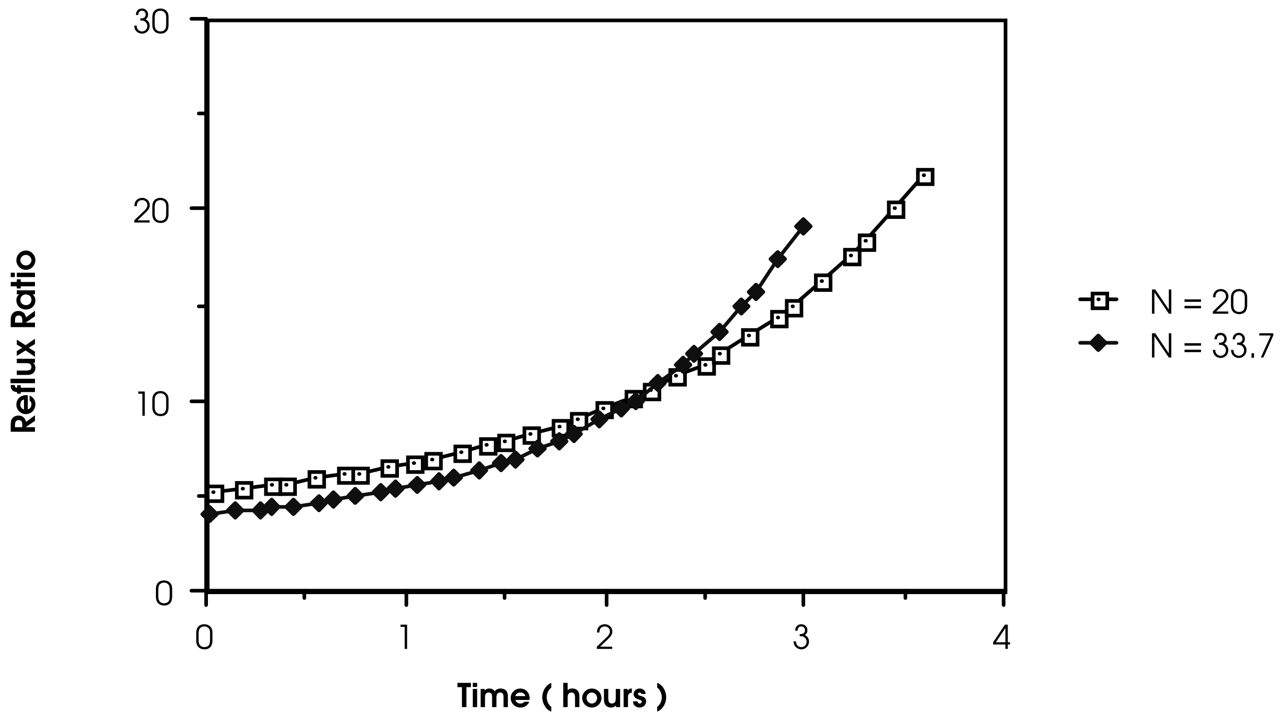
$$\text{Max } \{\text{Net Sales}(D, S_0)/(t_f + T_{set}) - \text{TAC}(N, V)\}$$

$N = 20$ trays, $T_{setup} = 1$ hour

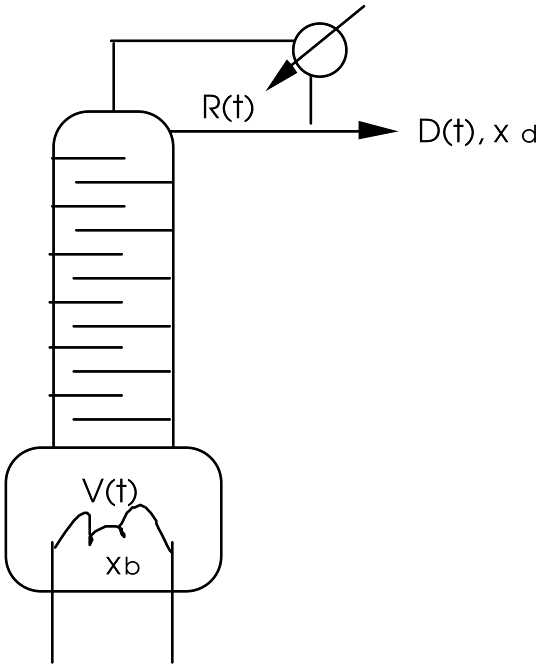
$x_d = 0.98$, $x_{feed} = 0.50$, $\alpha = 2$

$C_{prod}/C_{feed} = 4.1$

$V = 120$ moles/hr, $S_0 = 100$ moles.



Batch Distillation – Optimization Case Study - 2



$$\frac{dx_{1,N+1}}{dt} = \frac{V}{H_{N+1}}\left[y_{1,N} - x_{1,N+1}\right]$$

$$\frac{dx_{1,p}}{dt} = \frac{V}{H_p}\left[y_{1,p-1} - y_{1,p} + \frac{R}{R+1}\left(x_{1,p+1} - x_{1,p}\right)\right] \quad p = 1,\ldots,N$$

$$\frac{dx_{1,0}}{dt} = \frac{V}{S}\left[x_{1,0} - y_{1,0} + \frac{R}{R+1}\left(x_{1,1} - x_{1,0}\right)\right]$$

$$\frac{dD}{dt} = \frac{V}{R+1}$$

$$S^0 x^0_{i,0} = \left(S^0 - \sum_{p=1}^{N+1} H_p\right) x_{i,0} + \sum_{p=1}^{N+1} H_p\, x_{i,p}$$

$$\sum_1^C x_{i,p} = 1.0 \qquad \sum_1^C y_{i,p} = 1.0$$

<u>Ideal Equilibrium Equations</u>

$$y_{i,p} = K_{i,p}\ x_{i,p}$$

Binary Column (55/45, Cyclohexane, Toluene)
S_0 = 200, V = 120, H_p = 1, N = 10, ~8000 variables,
< 2 CPU hrs. (Vaxstation 3200)

Optimization Case Study - 2

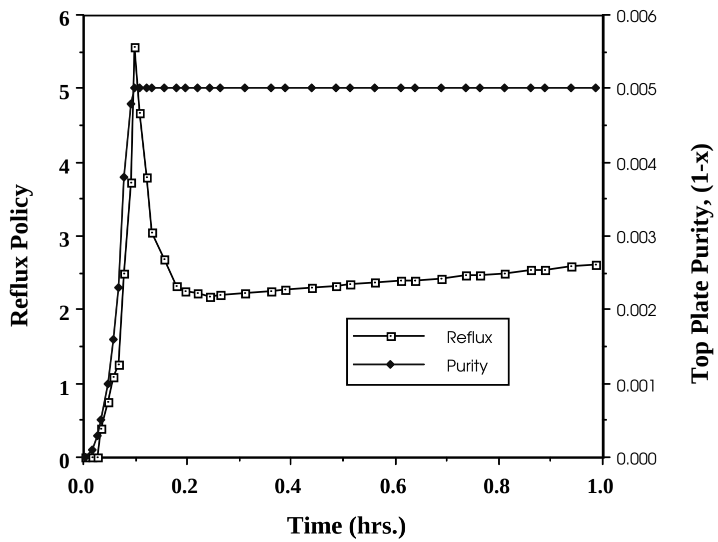
Modeling Assumptions

- Ideal Thermodynamics
- Constant Tray Holdup
- No Vapor Holdup
- Binary Mixture (55 toluene/45 cyclohexane)
- 1 hour operation
- Total Reflux Initial Condition

Cases Considered

- Constant Composition over Time
- Specified Composition at Final Time
- Best Constant Reflux Policy
- Piecewise Constant Reflux Policy

Reflux Optimization Cases



Constant Purity over Time

$x_1(t) \geq 0.995$

$D^*(t_f) = 38.61$

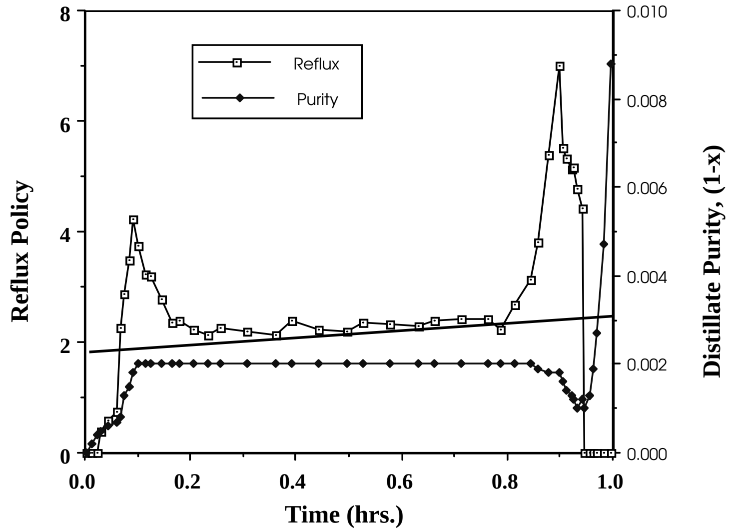
Overall Distillate Purity

$\int x_d(t)\, V/(R+1)\, dt) / D(t_f) \geq 0.998$

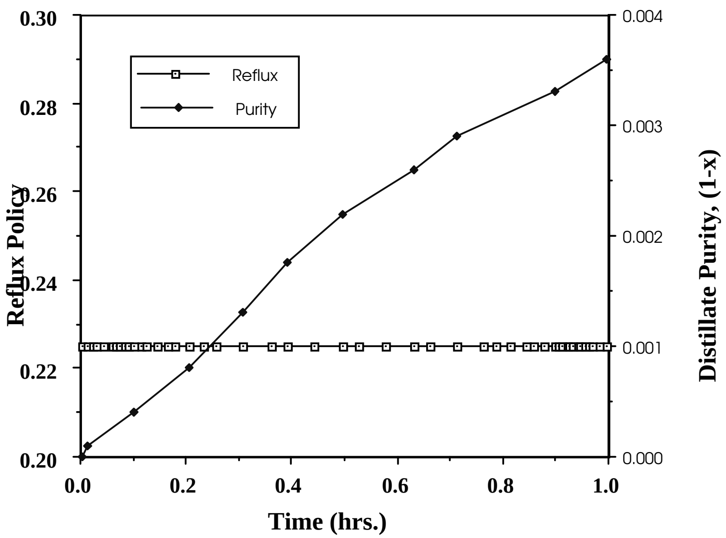
$D^*(t_f) = 42.34$

Shortcut Comparison

$D^*(t_f) = 37.03$



Reflux Optimization Cases



Constant Reflux over Time

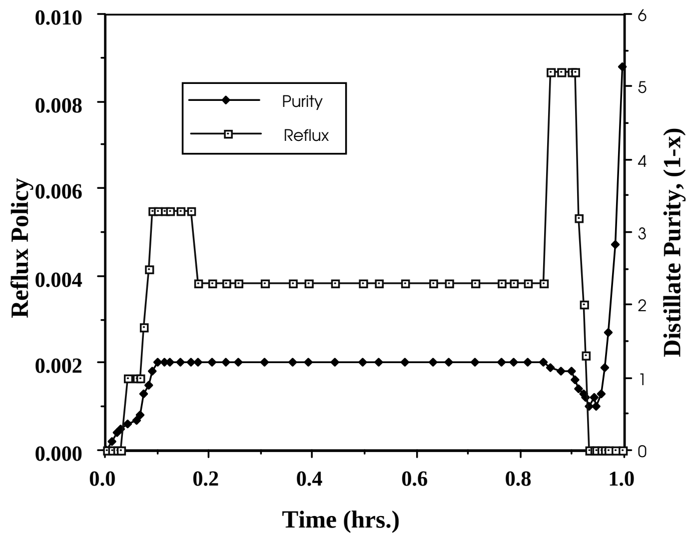
$x_d(t)\ V/(R+1)\ dt)\ /D(t_f)\ \geq 0.998$

$D^*(t_f) = 38.9$

Piecewise Constant Reflux over Time

$\int x_d(t)\ V/(R+1)\ dt)\ /D(t_f)\ \geq 0.998$

$D^*(t_f) = 42.26$

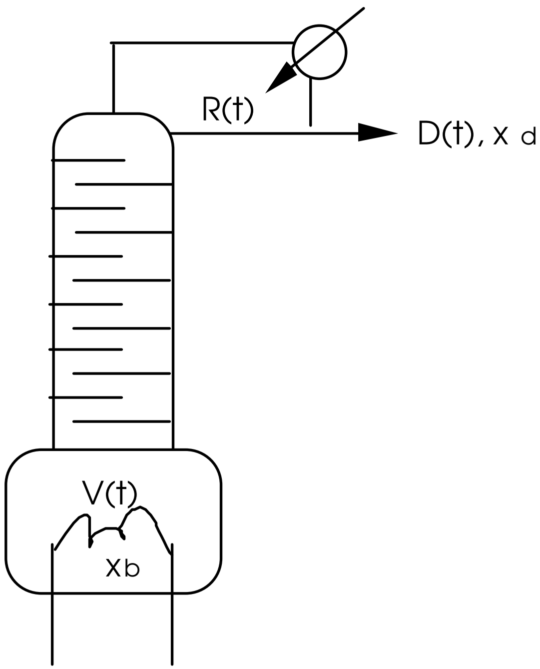


Batch Reactive Distillation – Case Study 3

Reversible reaction between acetic acid and ethanol

$$CH_3COOH + CH_3CH_2OH \leftrightarrow CH_3COOCH_2CH_3 + H_2O$$

t = 0, x = 0.25
for all components



$$\max \int_0^{t_f=1} D\,dt$$

s.t. DAE

$$x_D^{Ester} \geq 0.4600$$

Wajde & Reklaitis (1995)

Optimization Case Study - 3

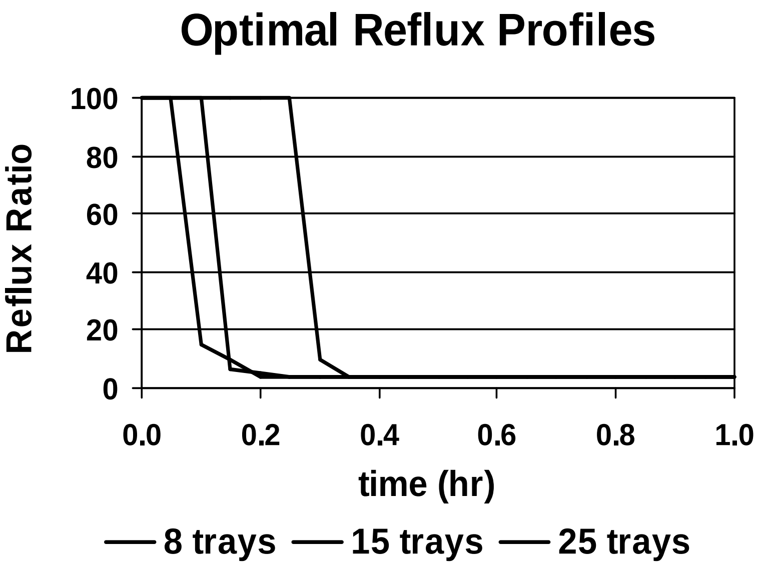
Modeling Assumptions

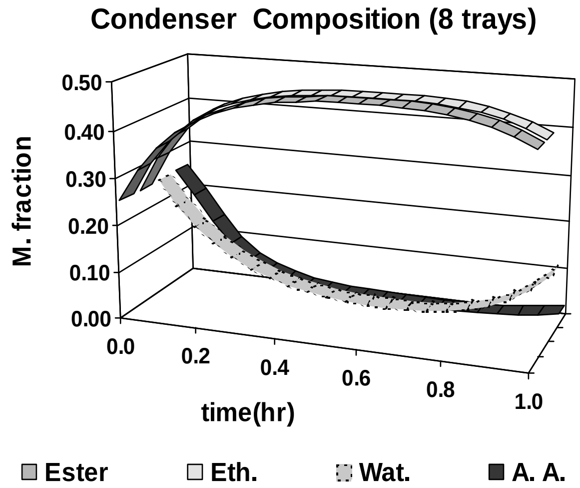
- Ideal Thermodynamics
- Constant Tray Holdup
- No Vapor Holdup
- Tertiary Mixture (EtOH, HOAc, ETAc, H_2O)
- Cold Start Initial Condition

Cases Considered

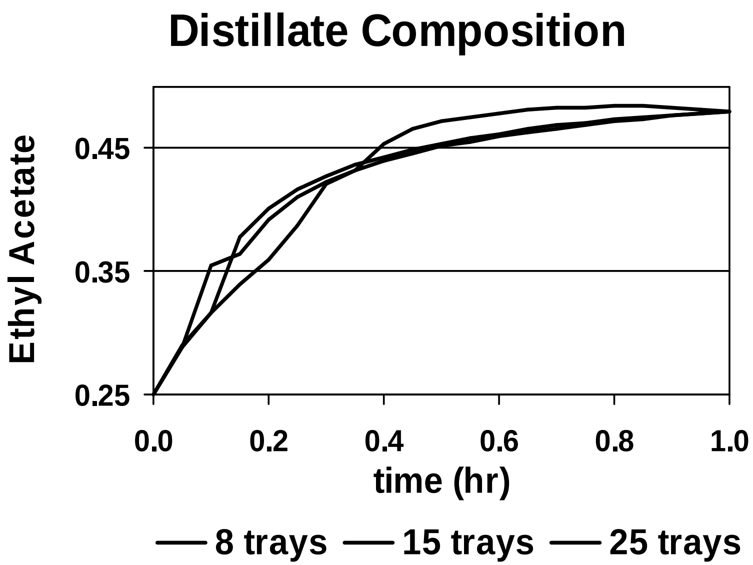
- Specified Composition at Final Time
- Optimum Reflux Policy
- Various Trays Considered (8, 15, 25)
- 1 hour operation

Batch Reactive Distillation



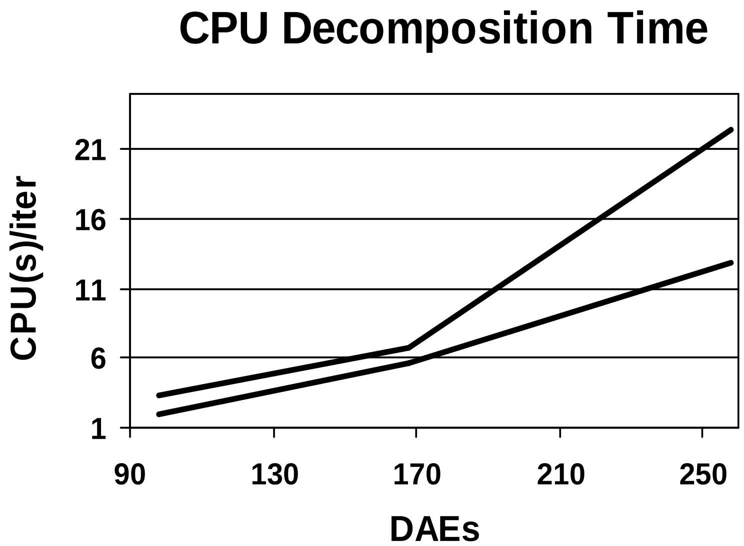


- < 5000 variables
- < 260 DAEs
- 10 degrees of freedom
- 10 finite elements
- < 50 IPOPT iterations
- < 11 CPU minutes

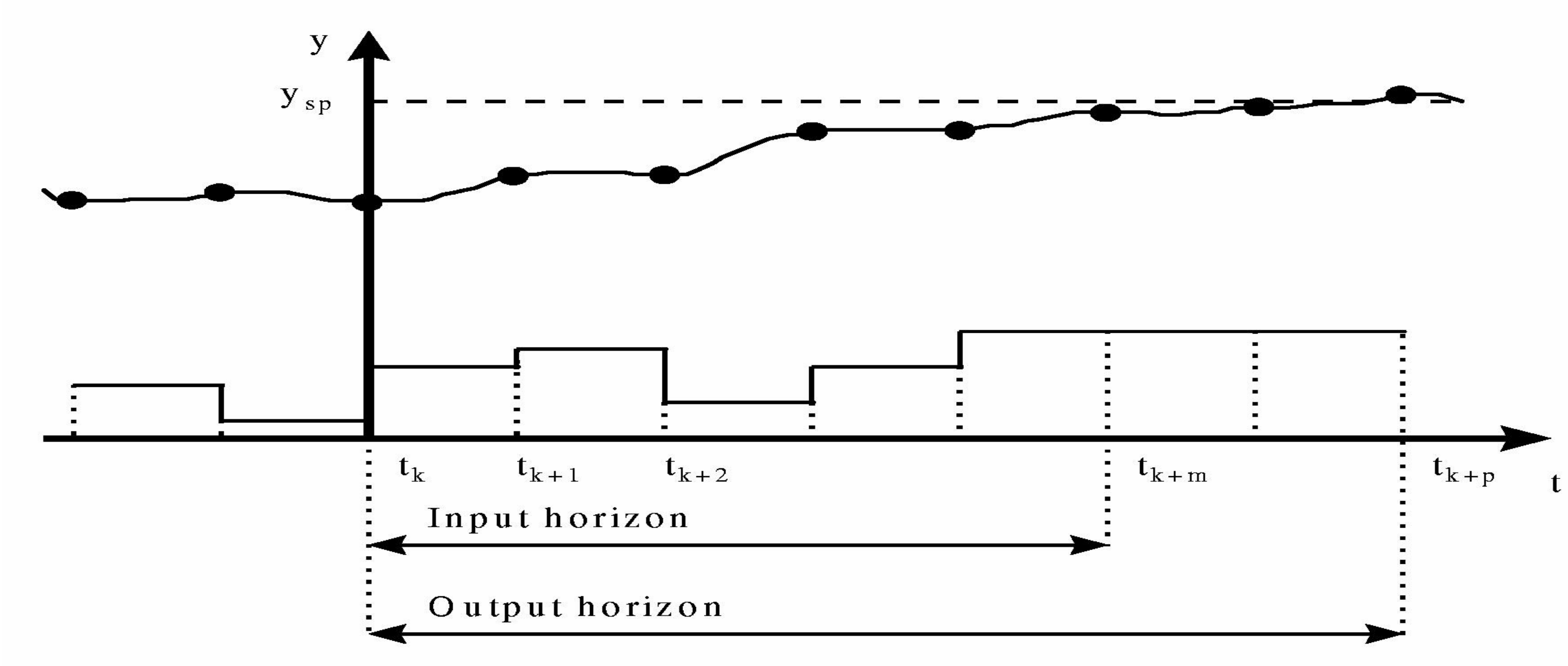


Batch Reactive Distillation

Trays	DAEs	Discretized Variables	Iterations	CPU (s) Global	CPU (s) Elemental
8	98	1788	14	56.4	37.2
15	168	3048	32	245.7	207.5
25	258	4678	45	1083.2	659.3



Nonlinear Model Predictive Control (NMPC)

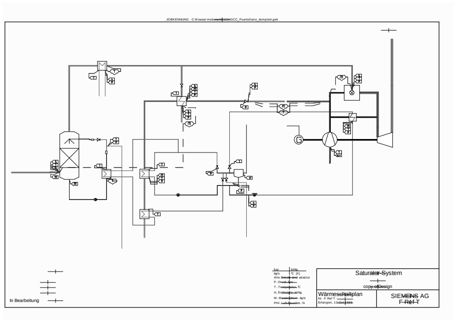


$$\min_{\boldsymbol{u}} \quad \sum \| y(t) - y^{\text{sp}} \|_{Q^y} + \sum \| \boldsymbol{u}(\boldsymbol{t}^k) - \boldsymbol{u}(\boldsymbol{t}^{k-1}) \|_{Q^u}$$

$$s.t. \quad \begin{aligned} & z'(t) = F(z(t), y(t), \boldsymbol{u}(t), t) \\ & 0 = G(z(t), y(t), \boldsymbol{u}(t), t) \\ & z(t) = z^{\text{init}} \\ & \text{Bound Constraints} \\ & \text{Other Constraints} \end{aligned}$$

Dynamic optimization in a MATLAB Framework

Dynamic Process



Dynamic Optimization Problem

Process Model

$$\mathbf{f}\left(\mathbf{x}', \mathbf{x}, \mathbf{y}, \mathbf{u}, \mathbf{p}, t\right) = \mathbf{0}$$

$$\boldsymbol{g}\left(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{u}, \boldsymbol{p}, t\right) = \mathbf{0}$$

Inequality Constraints

$$\mathbf{h}(\mathbf{x}', \mathbf{x}, \mathbf{y}, \mathbf{u}, \mathbf{p}, t) \le 0$$

Initial Conditions

$$\boldsymbol{x}(t_0) = \boldsymbol{x_0}$$

Constraints at Final Time

$$\varphi\left(\mathbf{x}'(t_f), \mathbf{x}(t_f), \mathbf{y}(t_f), \mathbf{u}(t_f), \mathbf{x}_0, \mathbf{p}, t_f\right) = 0$$

Objective Function

$$\min_{\mathbf{u}(t), \mathbf{p}, \mathbf{x}_0, t_f} \mathrm{P}\left(\mathbf{x}(t_f), \mathbf{y}(t_f), \mathbf{u}(t_f), \mathbf{p}, \mathbf{x}_0, t_f\right)$$

Discretization Method
No. of Time Elements
Collocation Order

Full Discretization of State and Control Variables

NLP Optimization Problem

Process Model

$$\hat{\mathbf{f}}\left(\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{u}}, \mathbf{p}, t_f, \mathbf{x_0}\right) = \mathbf{0}$$

$$\hat{\mathbf{g}}\left(\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{u}}, \mathbf{p}, t_f\right) = \mathbf{0}$$

Inequality Constraints

$$\hat{\mathbf{h}}(\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{u}}, \mathbf{p}, t_f) \le 0$$

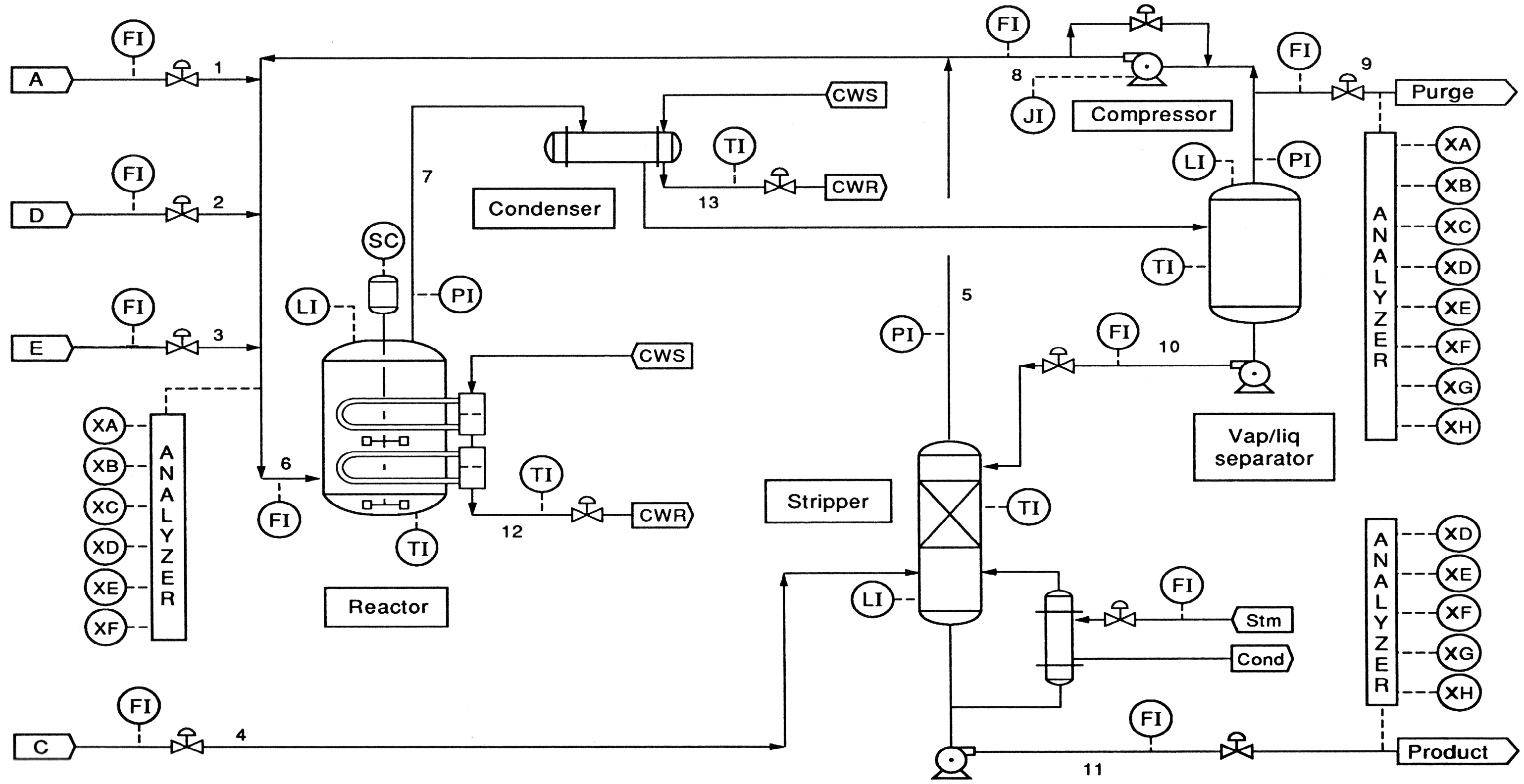
Constraints at Final Time

$$\varphi\left(\mathbf{x}_{N_t}, \mathbf{y}_{N_t}, \mathbf{u}_{N_t}, \mathbf{p}, t_f\right) = 0$$

Objective Function

$$\min \mathrm{P}\left(\mathbf{x}_{N_t}, \mathbf{y}_{N_t}, \mathbf{u}_{N_t}, \mathbf{p}, t_f\right)$$

Tennessee Eastman Process



Unstable Reactor

11 Controls; Product, Purge streams

Model extended with energy balances

Tennessee Eastman Challenge Process

DAE Model	
Number of differential equations	30
Number of algebraic variables	152
Number of algebraic equations	141
Difference (control variables)	11

NLP Optimization problem	
Number of variables of which are fixed	10920 0
Number of constraints	10260
Number of lower bounds	780
Number of upper bounds	540
Number of nonzeros in Jacobian	49230
Number of nonzeros in Hessian	14700

Method of Full Discretization of State and Control Variables

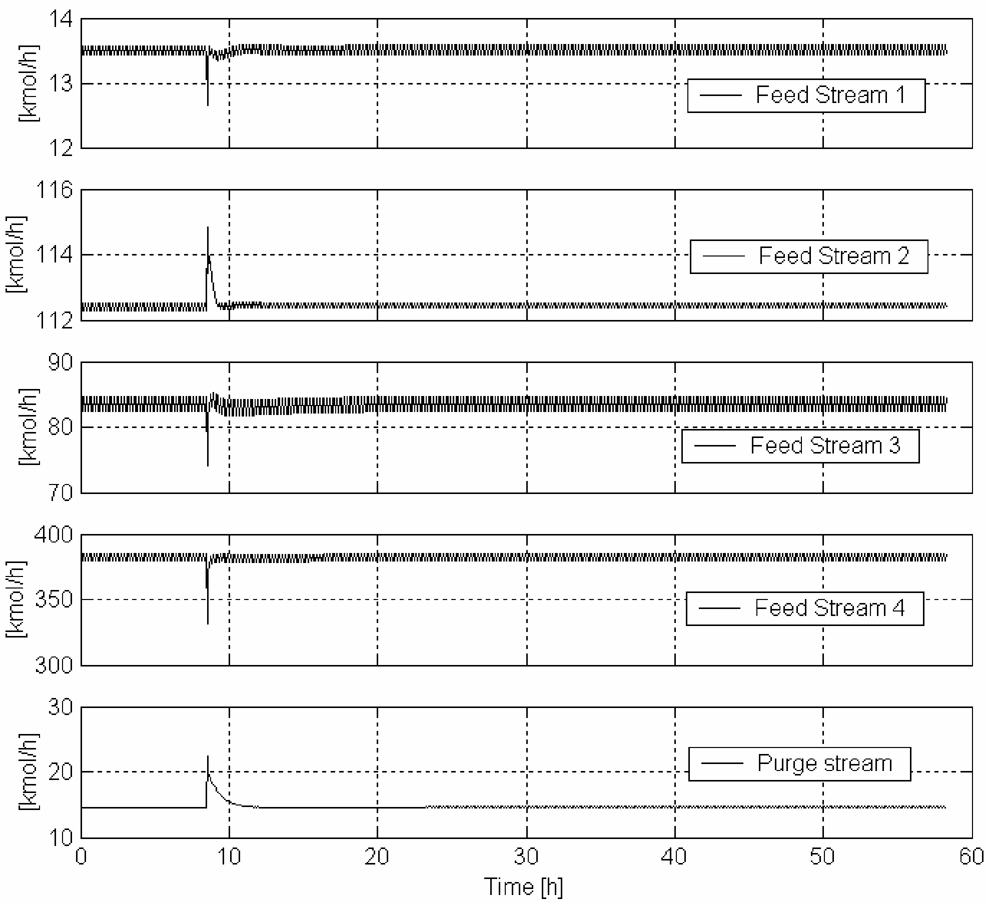
Large-scale Sparse block-diagonal NLP

Setpoint change studies

Process variable	Type	Magnitude
Production rate change	Step	-15% Make a step change to the variable(s) used to set the process production rate so that the product flow leaving the stripper column base changes from 14,228 to 12,094 kg h^{-1}
Reactor operating pressure change	Step	-60 kPa Make a step change so that the reactor operating pressure changes from 2805 to 2745 kPa
Purge gas composition of component B change	Step	+2% Make a step change so that the composition of component B in the gas purge changes from 13.82 to 15.82%

Setpoint changes for the base case [Downs & Vogel]

Case Study:
Change Reactor pressure by 60 kPa



Control profiles

All profiles return to their base case values

Same production rate

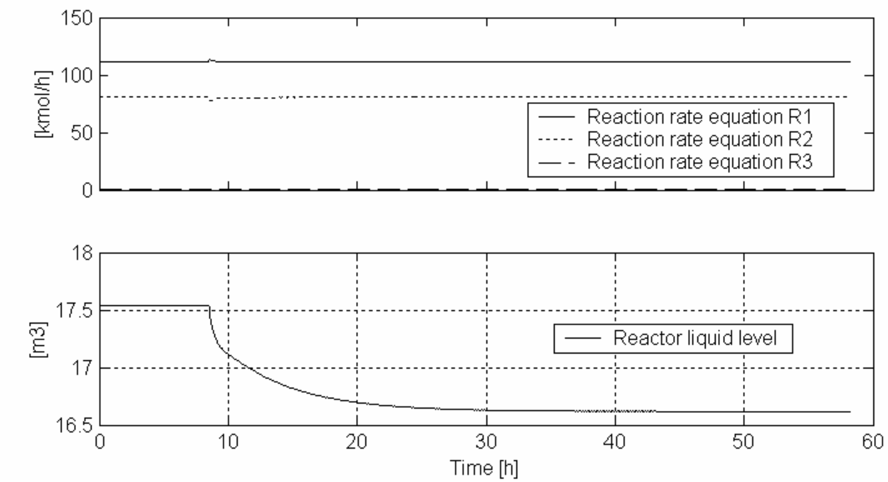
Same product quality

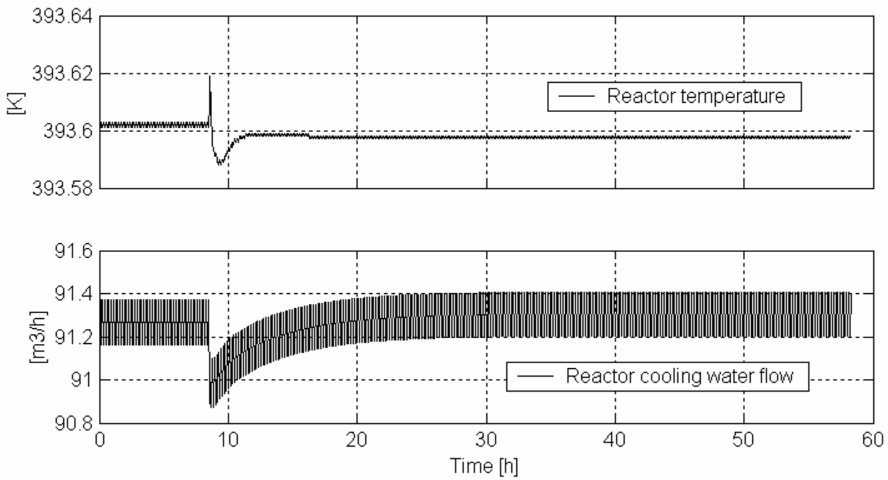
Same control profile

Lower pressure – leads to larger gas phase (reactor) volume

Less compressor load

TE Case Study – Results I





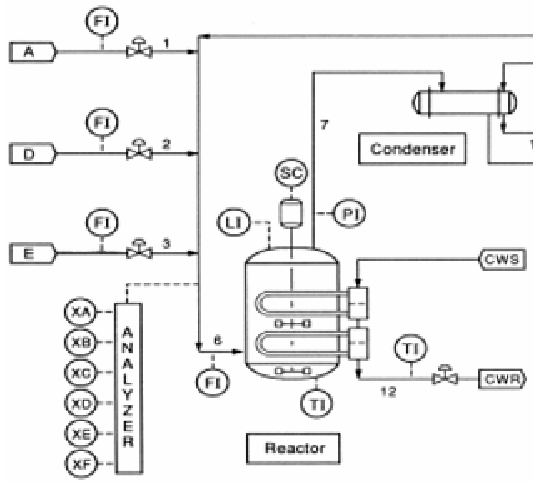
Shift in TE process

Same production rate

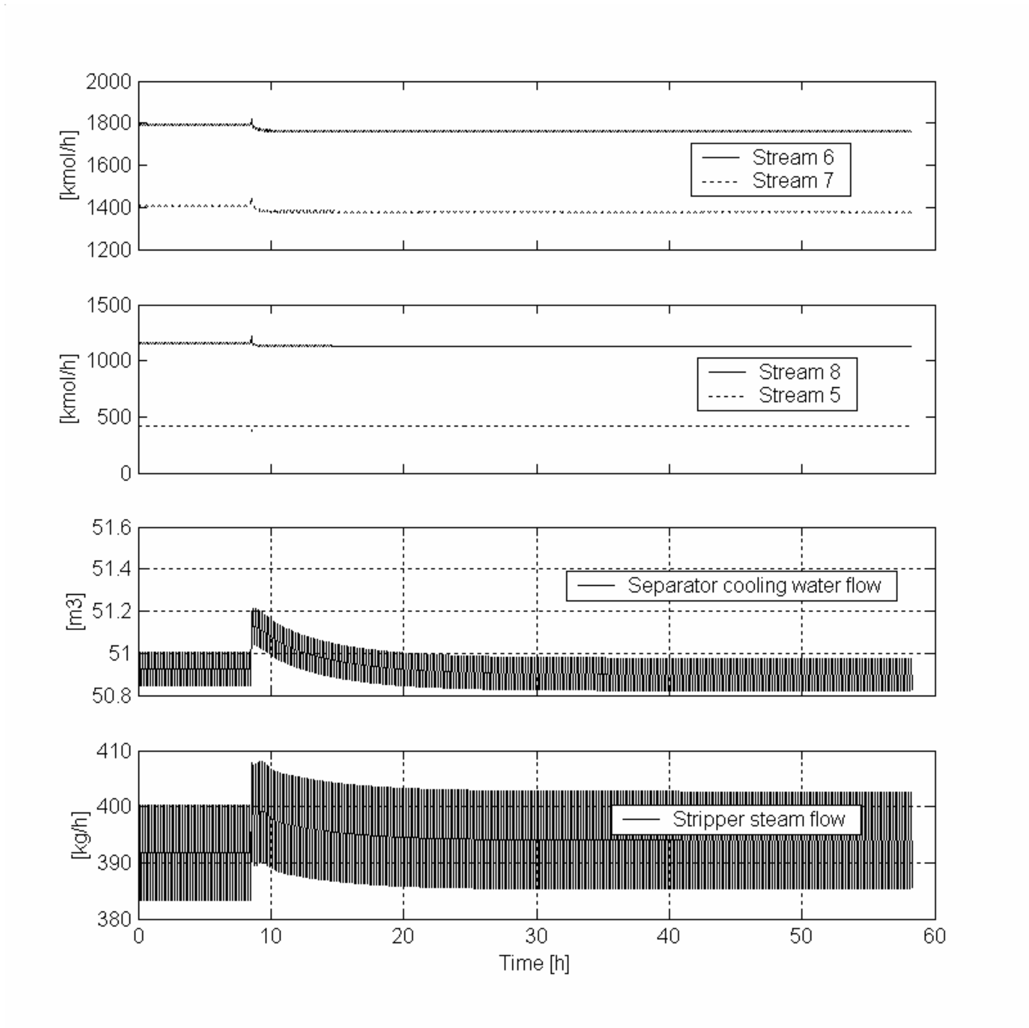
More volume for reaction

Same reactor temperature

Initially less cooling water flow (more evaporation)



Case Study- Results II



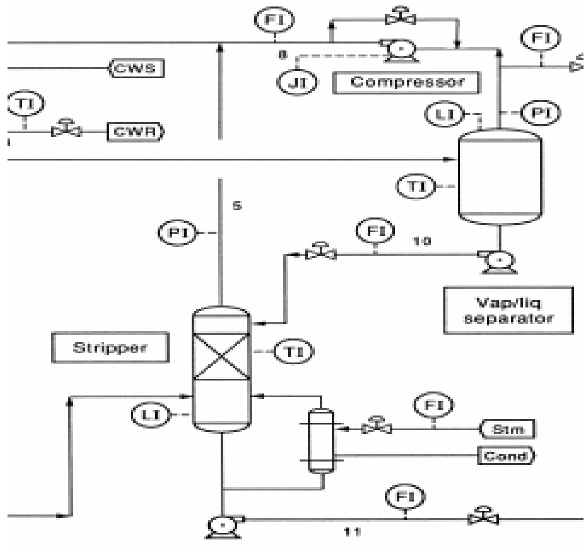
Shift in TE process

Shift in reactor effluent to more condensables

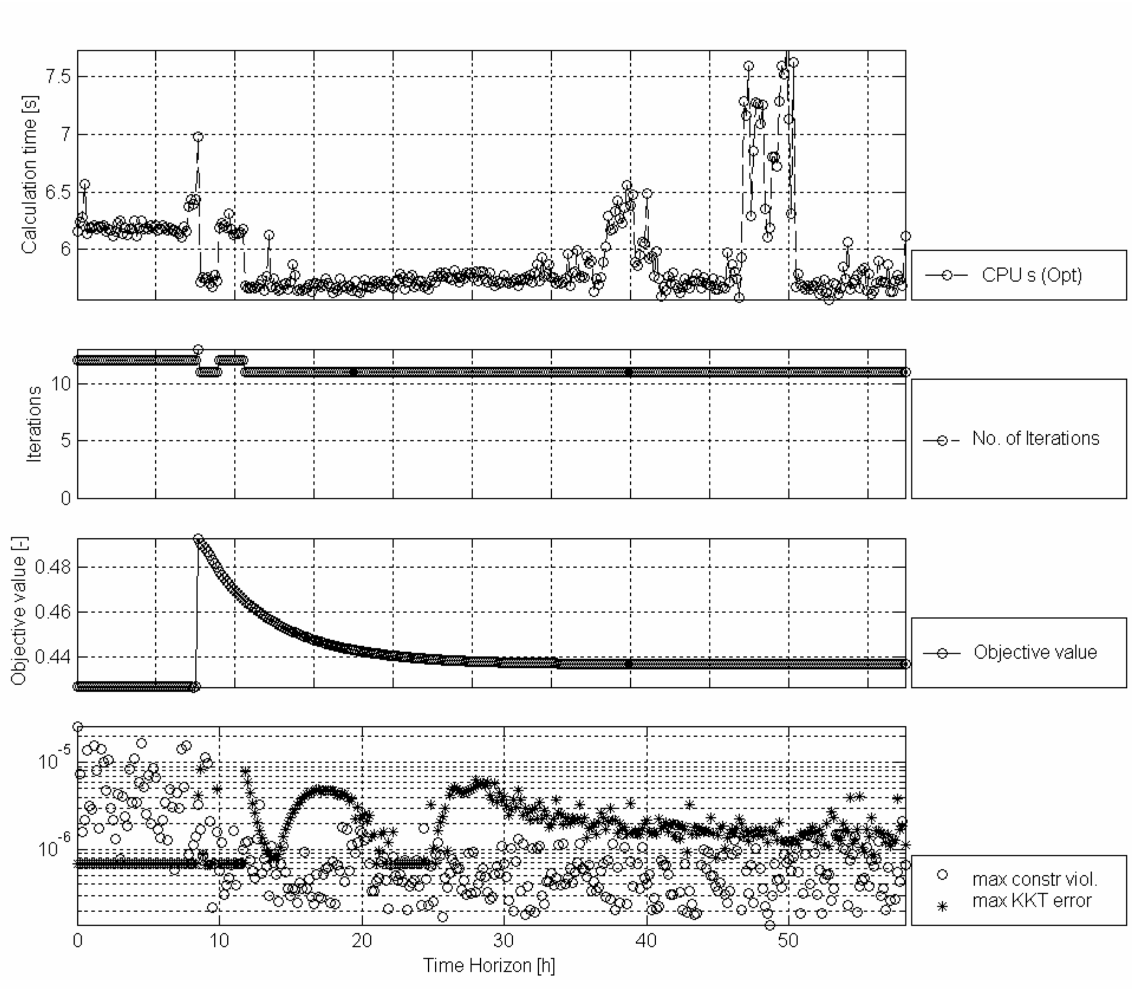
Increase cooling water flow

Increase stripper steam to ensure same purity

Less compressor work



Case Study:
Change Reactor Pressure by 60 kPa



Optimization with IPOPT

1000 Optimization Cycles

5-7 CPU seconds

11-14 Iterations

Optimization with SNOPT

Often failed due to poor conditioning

Could not be solved within sampling times

> 100 Iterations

Optimization as a Framework for Integration

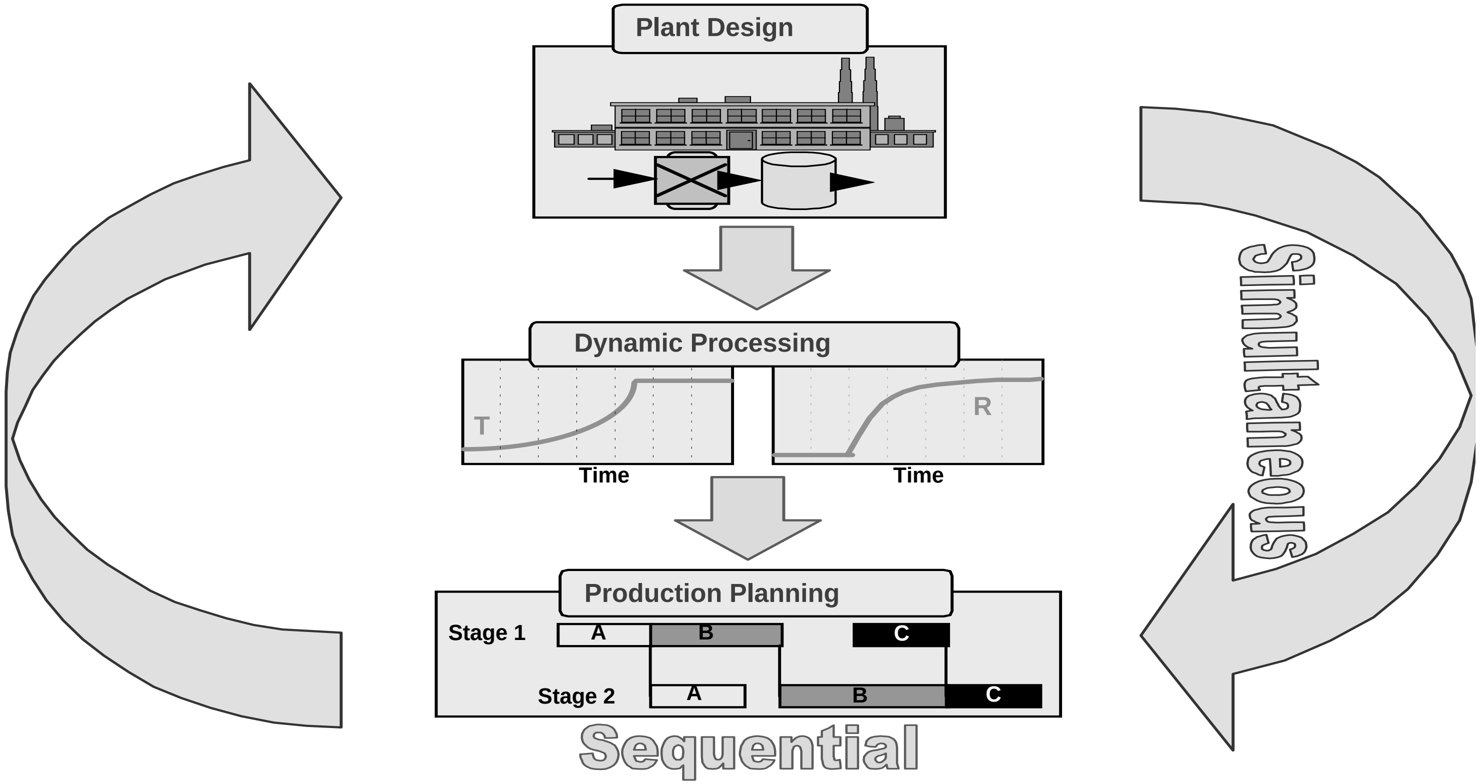
+ Directly handles interactions, multiple conditions
+ Trade-offs unambiguous, quantitative

- Larger problems to solve
- Need to consider a diverse process models

Research Questions

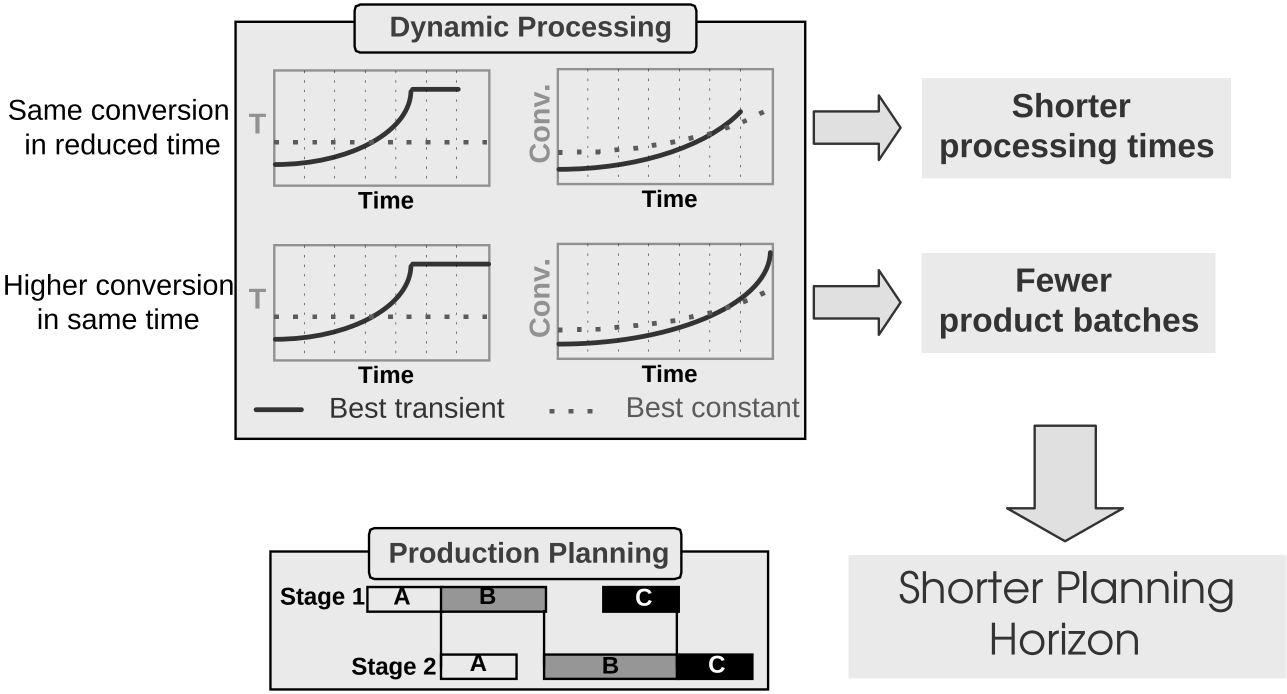
How should diverse models be integrated?
Is further algorithmic development needed?

Batch Integration Case Study



- What are the Interactions between Design and Dynamics and Planning?
- What are the differences between Sequential and Simultaneous Strategies?
- Especially Important in Batch Systems

Simultaneous Dynamic Optimization

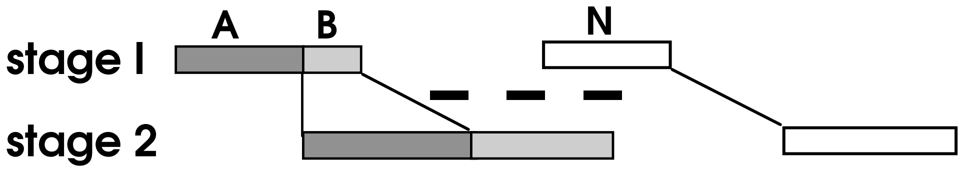


- discretize (DAEs), state and control profiles
- large-scale optimization problem
- handles profile constraints directly
- incorporates equipment variables directly
- DAE model solved only once
- converges for unstable systems

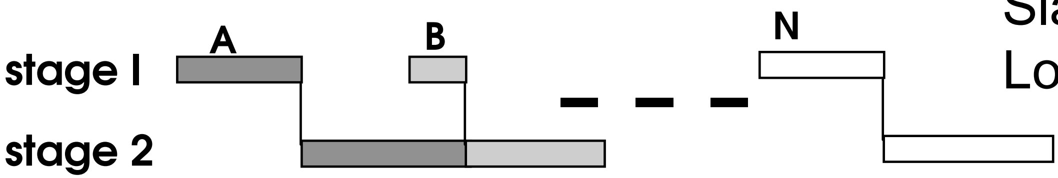
Scheduling Formulation

- sequencing of tasks, products equipment
- expensive discrete combinatorial optimization
- consider ideal transfer policies (UIS and ZW)
- closed form relations (Birewar and Grossmann, 1989)

<u>Unlimited Int. Storage(UIS)</u>
Short production cycle
Cycle time independent of sequence



<u>Zero Wait (ZW)</u>
Immediate transfer required
Slack times dependent on pair
Longer production cycle required



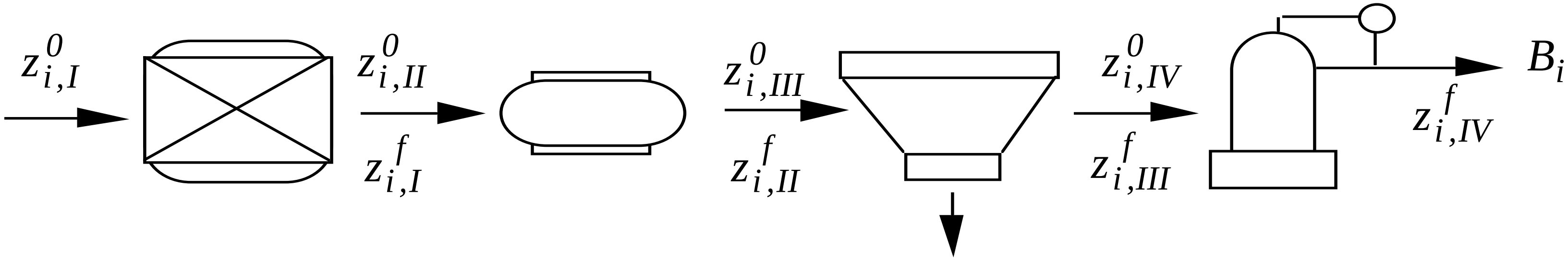
Case Study Example

4 stages, 3 products of different purity

Dynamic reactor - temperature profile

Dynamic column - reflux profile

$$A + B \rightarrow C$$
$$C + B \rightarrow P + E$$
$$P + C \rightarrow G$$



<u>Process Optimization Cases</u>

SQ - Sequential Design - Scheduling - Dynamics

SM - Simultaneous Design and Scheduling

Dynamics with endpoints fixed .

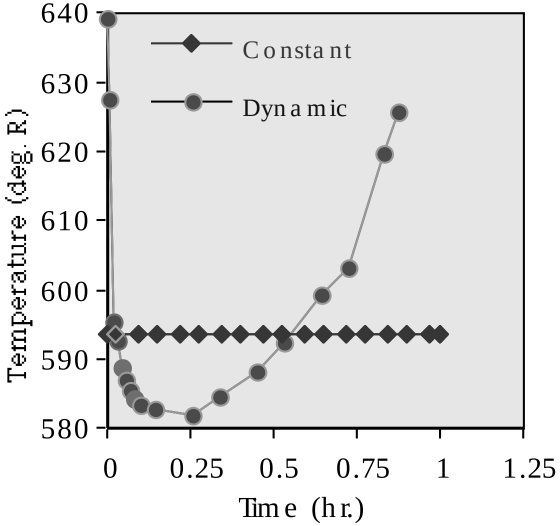
SM* - Simultaneous Design, Scheduling and Dynamics

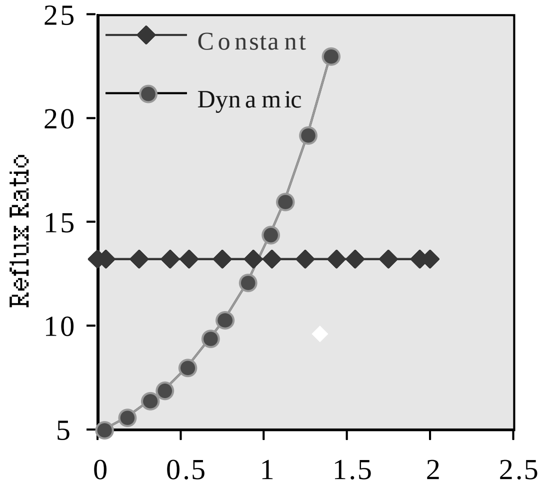
Scenarios in Case Study

Comparison of Dynamic vs. Best Constant Profiles

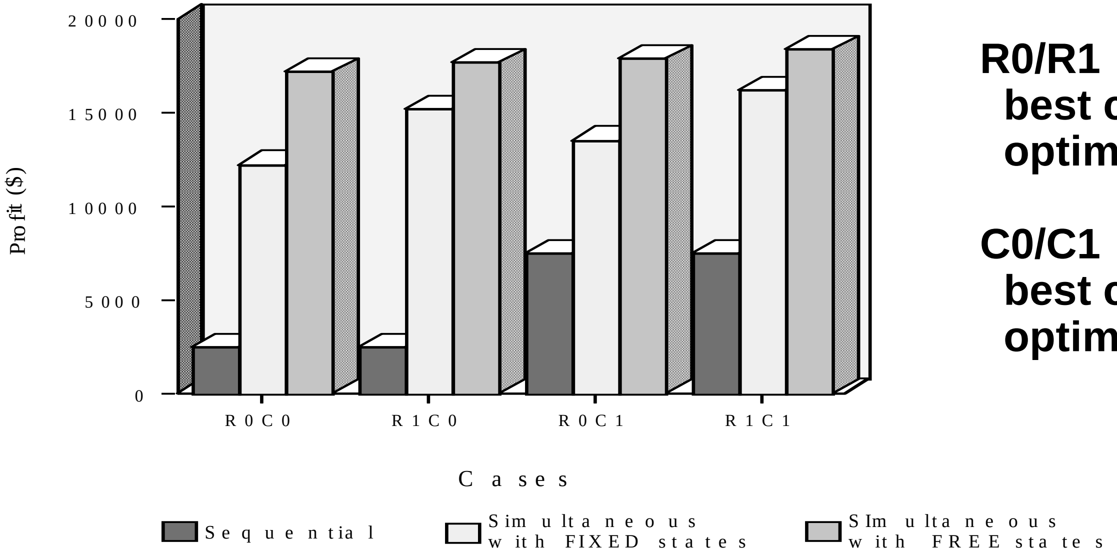
R0 - best constant temperature profile
R1 - optimal temperature policy

C0 - best constant reflux ratio
C1 - optimal reflux ratio



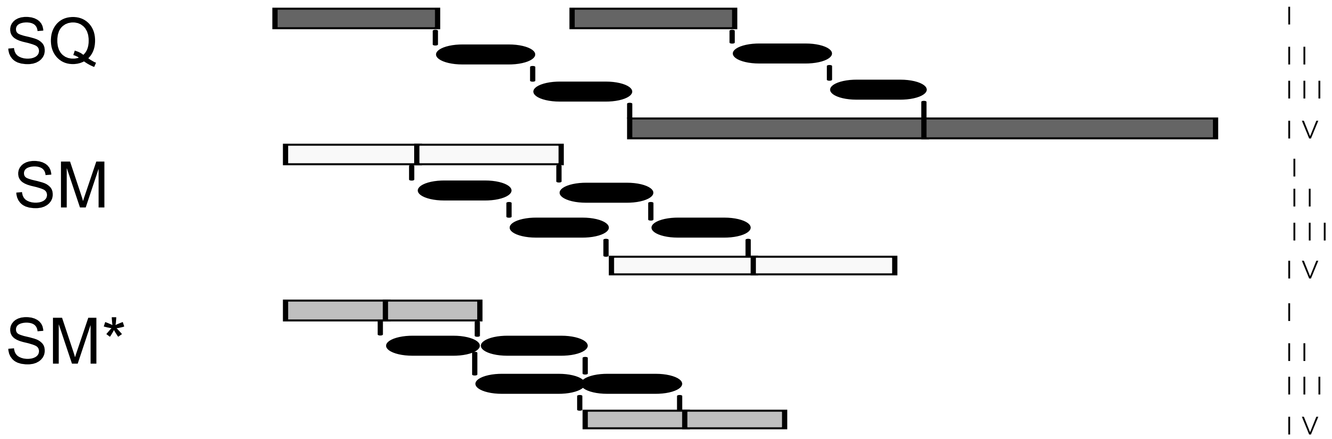


Results for Simultaneous Cases



R0/R1
best constant / optimal temperature

C0/C1
best constant / optimal reflux ratio



- ZW schedule becomes tighter

- less dependent on product sequences

Summary

Sequential Approaches

- Parameter Optimization
  - Gradients by: Direct and Adjoint Sensitivity Equations
- Optimal Control (Profile Optimization)
  - Variational Methods
  - NLP-Based Methods
- Require Repeated Solution of Model
- State Constraints are Difficult to Handle

Simultaneous Approach

- Discretize ODE's using orthogonal collocation on finite elements (solve larger optimization problem)
- Accurate approximation of states, location of control discontinuities through element placement.
- Straightforward addition of state constraints.
- Deals with unstable systems

Simultaneous Strategies are Effective

- Directly enforce constraints
- Solve model only once
- Avoid difficulties at intermediate points

Large-Scale Extensions

- Exploit structure of DAE discretization through decomposition
- Large problems solved efficiently with IPOPT

DAE Optimization Resources

References

Bryson, A.E. and Y.C. Ho, Applied Optimal Control, Ginn/Blaisdell, (1968).

Himmelblau, D.M., T.F. Edgar and L. Lasdon, Optimization of Chemical Processes, McGraw-Hill, (2001).

Ray. W.H., Advanced Process Control, McGraw-Hill, (1981).

Software

- Dynamic Optimization Codes

ACM – Aspen Custom Modeler

DynoPC - simultaneous optimization code (CMU)

COOPT - sequential optimization code (Petzold)

gOPT - sequential code integrated into gProms (PSE)

MUSCOD - multiple shooting optimization (Bock)

NOVA - SQP and collocation code (DOT Products)

- Sensitivity Codes for DAEs

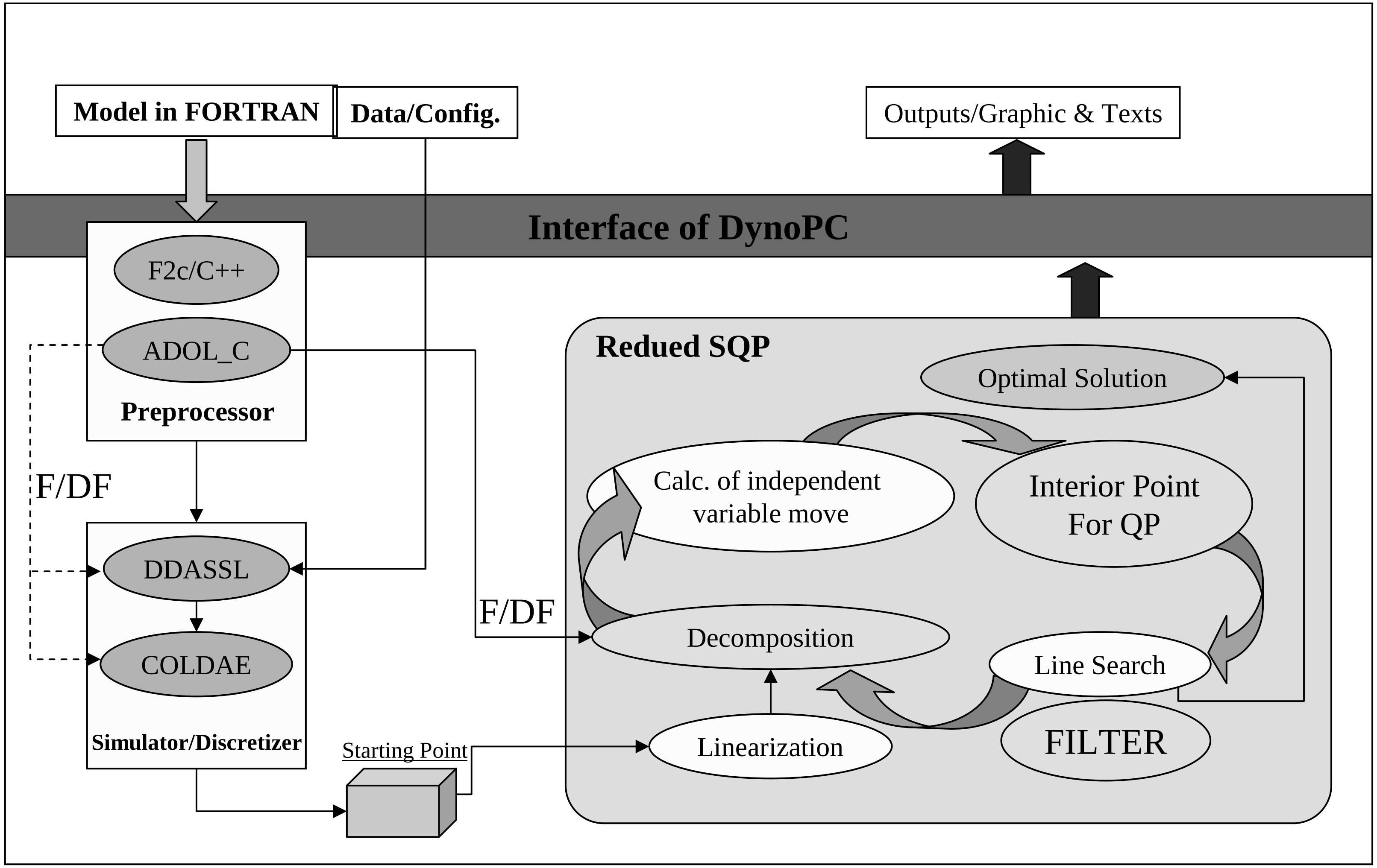
DASOLV - staggered direct method (PSE)

DASPK 3.0 - various methods (Petzold)

SDASAC - staggered direct method (sparse)

DDASAC - staggered direct method (dense)

DynoPC – Windows Implementation



Example: Batch Reactor Temperature

$$A \xrightarrow{1} B \xrightarrow{2} C$$

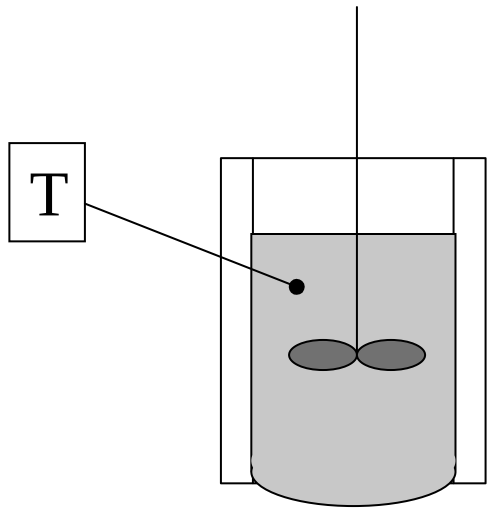
$$Max \quad b(t_f)$$

$s.t.$

$$\frac{da}{at} = -k_1 \exp\left(-\frac{E_1}{RT}\right) \cdot a$$

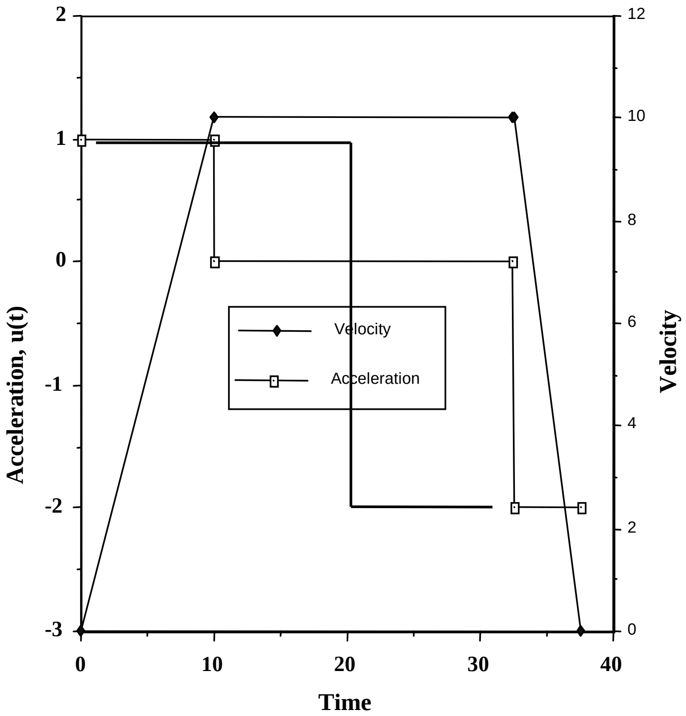
$$\frac{db}{at} = k_1 \exp\left(-\frac{E_1}{RT}\right) \cdot a - k_2 \exp\left(-\frac{E_2}{RT}\right) \cdot b$$

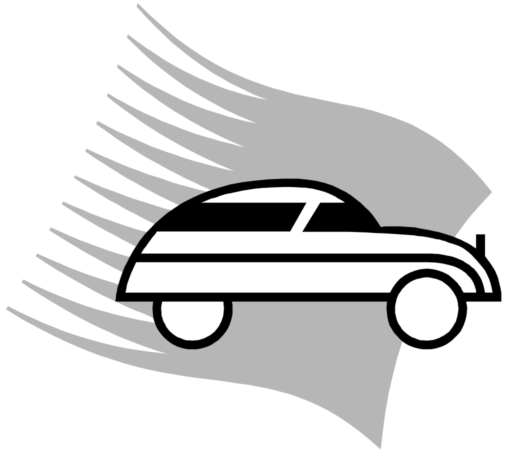
a + b + c = 1



Example: Car Problem

$$\begin{aligned} \text{Min} \quad & t_f \\ \text{s.t.} \quad & z_1' = z_2 \\ & z_2' = u \\ & z_2 \le z_{max} \\ & -2 \le u \le 1 \end{aligned}$$





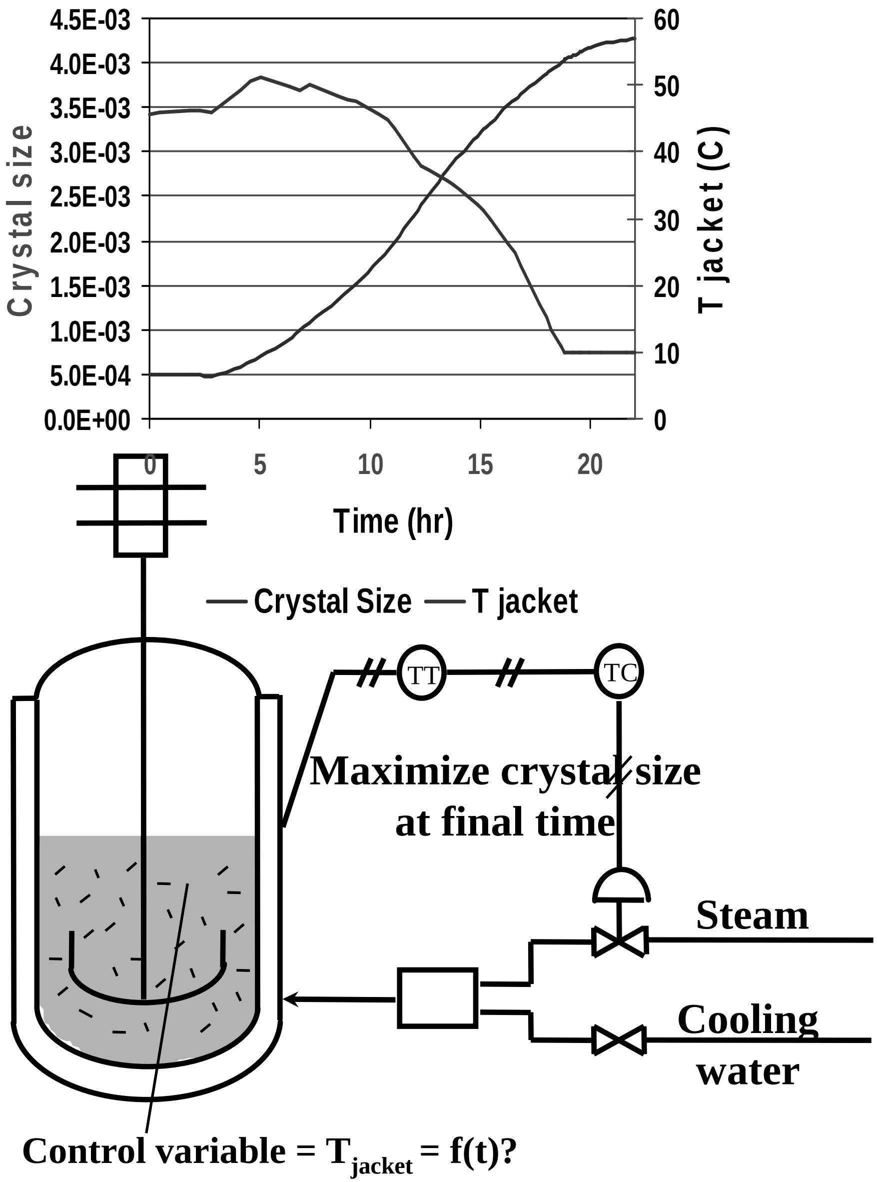
```
subroutine model(nz,ny,nu,np,t,z,dmz,y,u,p,f)
double precision t, z(nz),dmz(nz), y(ny),u(nu),p(np)

double precision f(nz+ny)

f(1) = p(1)*z(2) - dmz(1)
f(2) = p(1)*u(1) - dmz(2)

return
end
```

Example: Crystallizer Temperature




```
SUBROUTINE model(nz,ny,nu,np,x,z,dmz,y,u,p,f)
implicit double precision (a-h,o-z)
double precision f(nz+ny),z(nz),dmz(nz),Y(ny),yp(4),u(1)
double precision kgr, ln0, ls0, kc, ke, kex, lau, deltT, alpha
dimension a(0:3), b(0:3)
data alpha/1.d-4/,a/-66.4309d0, 2.8604d0, -.022579d0, 6.7117d-5/,
+  b/16.08852d0, -2.708263d0, .0670694d0, -3.5685d-4/, kgr/ 4.18d-3/,
+  en / 1.1d0/, ln0/ 5.d-5/, Bn / 3.85d2/, em / 5.72/, ws0/ 2.d0/,
+  Ls0/ 5.d-4     /, Kc / 35.d0 /, Kex/ 65.d0/, are/ 5.8d0 /,
+  amt/ 60.d0  /, V0 / 1500.d0/, cw0/ 80.d0/,cw1/ 45.d0/,v1 /200.d0/,
+                    tm1/ 55.d0/,x6r/0.d0/, tem/ 0.15d0/,clau/ 1580.d0/,lau/1.35d0/,
+   cp/ 0.4d0    /,cbata/ 1.2d0/, calfa/ .2d0   /, cwt/ 10.d0/

ke = kex*area
x7i = cw0*lau/(100.d0-cw0)
v = (1.d0 - cw0/100.d0)*v0
w = lau*v0
yp(1) = (deltT + dsqrt(deltT**2 + alpha**2))*0.5d0
yp(2) = (a(0) + a(1)*yp(4) + a(2)*yp(4)**2 + a(3)*yp(4)**3)
yp(3) = (b(0) + b(1)*yp(4) + b(2)*yp(4)**2 + b(3)*yp(4)**3)
deltT = yp(2) - z(8)
yp(4) = 100.d0*z(7)/(lau+z(7))

f(1) = Kgr*z(1)**0.5*yp(1)**en - dmz(1)
f(2) = Bn*yp(1)**em*1.d-6 - dmz(2)
f(3) = ((z(2)*dmz(1) + dmz(2) * Ln0)*1.d+6*1.d-4) - dmz(3)
f(4) = (2.d0*cbata*z(3)*1.d+4*dmz(1)+dmz(2)*Ln0**2*1.d+6)-dmz(4)
f(5) = (3.d0*calfa*z(4)*dmz(1)+dmz(2)*Ln0**3*1.d+6) - dmz(5)
f(6) = (3.d0*Ws0/(Ls0**3)*z(1)**2*dmz(1)+clau*V*dmz(5))-dmz(6)
f(7) = -dmz(6)/V - dmz(7)
f(8) = (Kc*dmz(6) - Ke*(z(8) - u(1)))/(w*cp) - dmz(8)
f(9) = y(1)+YP(3)- u(1)
return
end
```