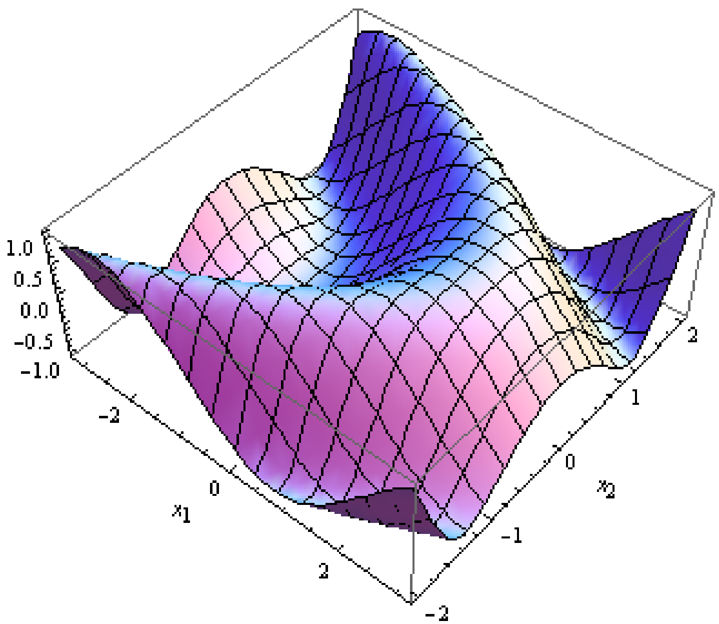
CONTROL ROOM ACCELERATOR PHYSICS

Day 3

Mathematical Programming and the XAL Solver

Outline

1. Introduction
2. Quantizing the Optimum
3. Mathematical Programming
4. Optimization
5. Solving Nonlinear Equations



Mathematical Programming

Introduction: Real World Problem Formulation

- Goals, Objectives, and Figure-of-Merit (FOM)
  - Goal - Want to make something happen
    - E.g. minimize orbit deviation
  - Figure of merit (FOM) quantizes progress of this goal
  - Figure of merit also called “objective” in mathematical programming

- Variables
  - What you vary to achieve your goal (affecting the figure-of-merit)
    - E.g. dipole corrector strengths
  - Usually these have limits (power supply capabilities)

- Constraints
  - Variables can only live in a certain region of parameter space
    - E.g. keep the orbit deviation at some point fixed in order to go around an obstruction

Mathematical Programming

Introduction: Constrained Optimization

- One way of implementing constraints is through penalty functions
  - Optimization is penalized when variables go out of bounds

- Constrained optimization versus penalty functions
  - Some packages include constrained optimization directly
  - Otherwise you can augment the FOM with a penalty function
    - E.g. FOM = Cost + Penalty where
      penalty = 1 – (bending field/existing magnet capability)^2
  - This has drawbacks though (the terms can fight each other)

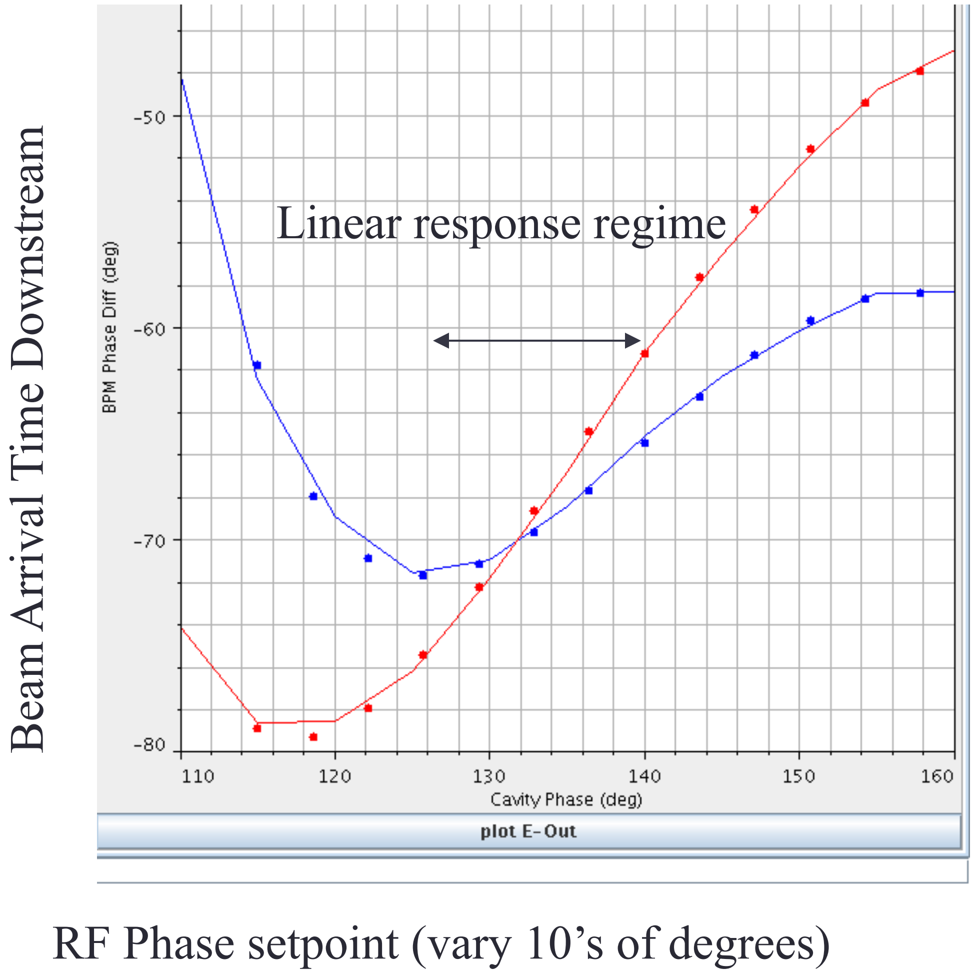
- Most optimization packages minimize the FOM,
  - If you need a maximization just use 1/FOM

- Variables
  - Usually the user provides a list of the variables and their limits

Mathematical Programming
Introduction: Non-linear Constrained Optimization

- Great for solving real world problems
- You don't need to know any math! (well, a little)
- In years past with slower processors, many techniques involved using advanced mathematical techniques – appropriate for the particular application
- Now-days a sledge hammer works fine
- Open XAL has such a sledge hammer in its toolbox (the "Solver")

Example of a Non-linear Solver Application



- XAL RF Phase setting application (PASTA)
- Large variations of the RF phase result in non-linear effects on the beam

Mathematical Programming

Overview

- Most problems involving optimization and/or the solution of nonlinear equations can be put into the framework of *mathematical programming*.
  - Usually we have several free parameters (e.g., magnet strengths) - the vector $\mathbf{x}$ represents these parameters in the vector space where we are looking for solutions (typically $\mathbf{R}^n$)
  - We take an initial "guess" for the solution $\mathbf{x}_0$
  - Using an (intelligent?) algorithm we iteratively update the current value of $\mathbf{x}_i$ to $\mathbf{x}_{i+1}$ usually with a policy of the form

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i \mathbf{d}$$

where $\mathbf{d}_i$ is the search direction and α_i is the search length at the i^{th} iteration.

Mathematical Programming

Overview (cont.)

- The method by which we chose the search direction $\mathbf{d}_i$ identifies the algorithm. (Still a topic of current research.)

- Some of the more popular are…
  - Newton (Ralphson) – simple technique based on derivatives
  - Conjugate gradients – the "expanding subspace" theorem
  - GMRES – Generalized minimal residual (reducing res. error)
  - Simplex – Inspection of constraint vertices
  - Genetic Algorithms – Analogous to genetic base pair expression
  - Dynamic Programming – Hamilton-Jacobi-Bellman equation

- For example, when using the Newton method to minimize a functional $J(\mathbf{x})$, the search directions are picked in the direction opposite to the gradient $\nabla J(\mathbf{x})$

Mathematical Programming

Overview (cont.)

- Many of these algorithms are "canned" in mathematical software packages
  - Consequently they are easy to employ
  - In order to use one of these canned mathematical programming packages (for equation solving, or for optimization), we need to formulate our problem as a mathematical programming problem.

- For example, nonlinear optimization is a basic mathematical programming application
  - Basic (unconstrained) minimization problem

  Given a functional $J : \mathbf{R}^n \to \mathbf{R}$,

$$\text{find } \bar{\mathbf{x}} \in \mathrm{R}^n \quad \text{such that} \quad J(\bar{\mathbf{x}}) \leq J(\mathbf{x}) \text{ for all } \mathbf{x} \in \mathrm{R}^n$$

$$\textit{or} \quad \bar{\mathbf{x}} = \arg\min_{\mathrm{x} \in \mathrm{R}^n} J(\mathbf{x})$$

J is the Figure of Merit

Mathematical Programming

A Warning on Algorithms

- Some mathematical programming algorithms rely upon the smoothness of the objective $J(\mathbf{x})$
  - These algorithms tend to use derivative information to compute the $\{\mathbf{d}_i\}$
  - Taking derivatives of noisy data can lead to problems – the noise component is usually amplified

- When working with parameters $\mathbf{x}$ obtained from experimental data it may be wise to avoid the so-called *descent* algorithms that typically employ the gradient of $J(\mathbf{x})$ (at least approximately). Instead, try algorithms using direct evaluation…
  - Genetic algorithms
  - Simplex algorithms
  - Etc.

- Note, however, repeated direct evaluation can be expensive!

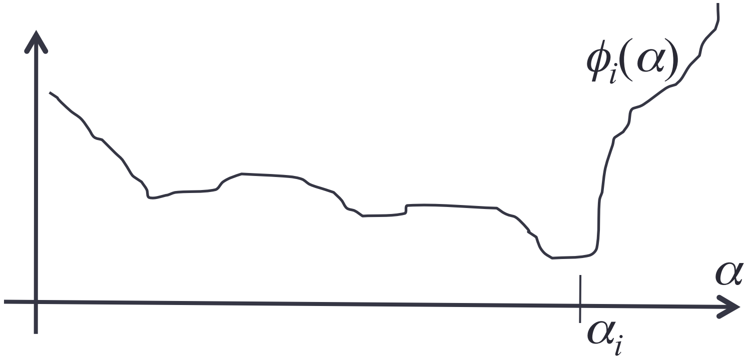
Mathematical Programming

Example: Function Minimization via Newton

Newton minimization: Newton minimization is arguably the most simple descent-type algorithm where the search directions are picked as $-\nabla J(\mathbf{x}_i)$

- For any point $\mathbf{x}_i$, the gradient $-\nabla J(\mathbf{x}_i)$ gives search direction $\mathbf{d}_i$
- The search length α_i is determined through a separate line search algorithm which minimizes the scalar function

$$\phi_i(\alpha) \equiv J(\mathbf{x}_i + \alpha \mathbf{d}_i)$$



- Thus we have

$$\mathbf{d}_i = -\nabla J(\mathbf{x}_i) = -\begin{pmatrix} \partial J / \partial x_1 \\ \vdots \\ \partial J / \partial x_n \end{pmatrix}$$

$$\alpha_i = \arg\min_{\alpha} \phi_i(\alpha)$$

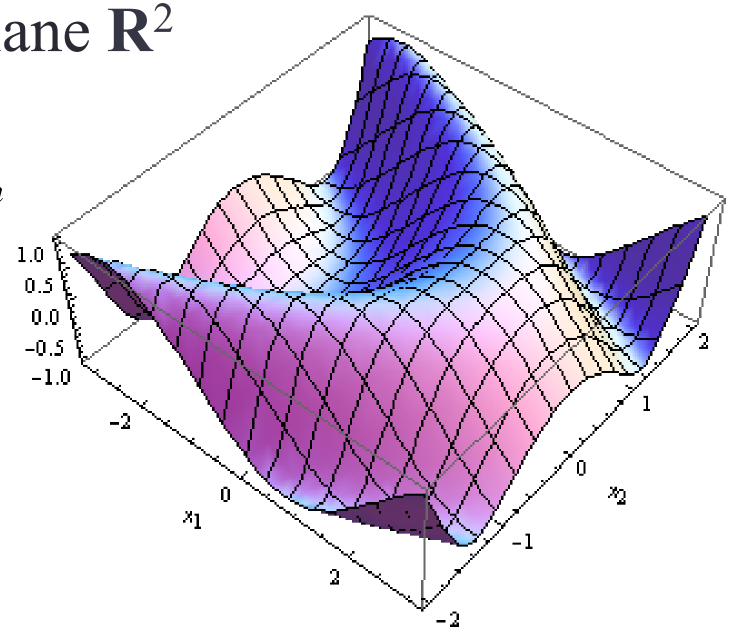
$$\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i \mathbf{d}_i$$

Mathematical Programming

Example (cont.): Function Minimization via Newton

Consider the nonlinear functional on the plane $\mathbf{R}^2$

$$J(\mathbf{x}) = \sin\left(x_1 + x_2^2\right) \quad \text{where} \quad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \in \mathbf{R}^n$$



- For any point $\mathbf{x}$, the gradient $\nabla J(\mathbf{x})$ gives $-\mathbf{d}$

$$\mathbf{d}_i = -\nabla J(\mathbf{x}_i) = -\begin{pmatrix} \cos(x_1 + x_2^2) \\ 2x_2 \cos(x_1 + x_2^2) \end{pmatrix}$$

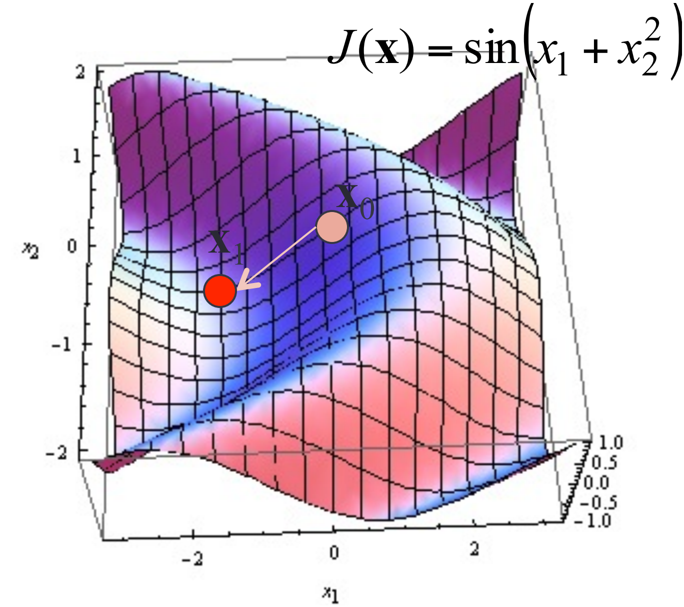
$$\alpha_i = \arg\min_{\alpha} \sin\left[\left(x_1 - \alpha\cos(x_1 + x_2^2)\right) + \left(x_2^2 - \alpha 2x_2 \cos(x_1 + x_2^2)\right)^2\right]$$

Mathematical Programming

Example (cont.): Function Minimization via Newton

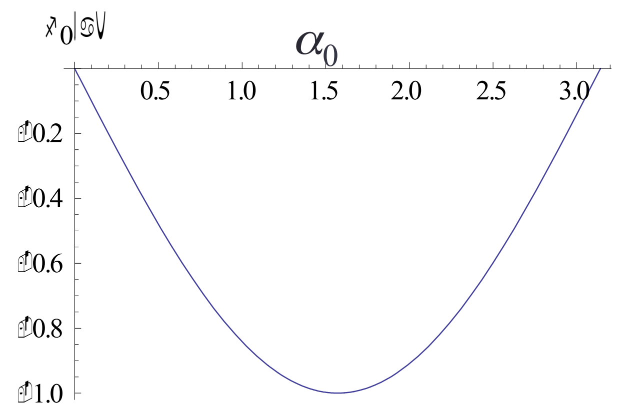
- For example, starting at $\mathbf{x}_0 = (0,0)$

$$\left.\begin{array}{l}\alpha_0 = \pi/2 \\ \mathbf{d}_0 = \begin{pmatrix}-1 \\ 0\end{pmatrix}\end{array}\right\} \Rightarrow \mathbf{x}_1 = \begin{pmatrix}-\pi/2 \\ 0\end{pmatrix}$$



- For the next iterate we compute

$$\mathbf{d}_1 = \begin{pmatrix}0 \\ 0\end{pmatrix}$$



After which $\mathbf{x}_i = (-\pi/2,0)$ for $i > 1$

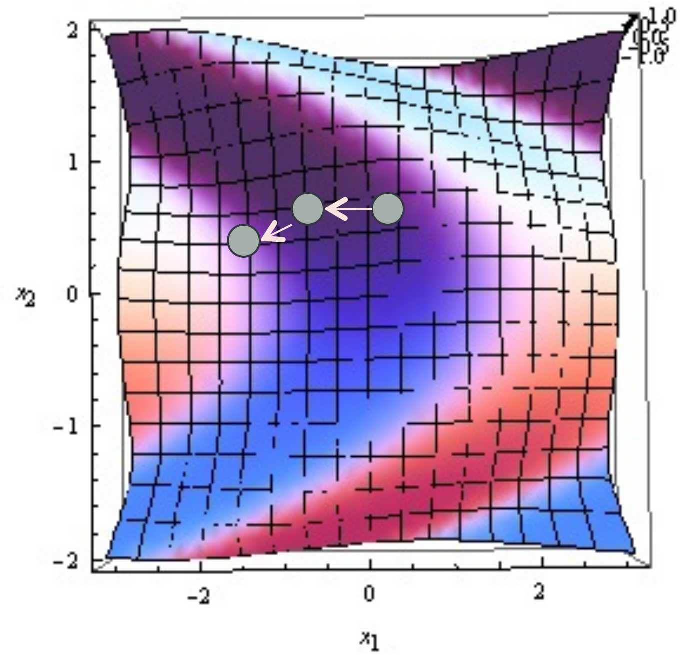
Mathematical Programming

Example (cont.): Function Minimization via Newton

- However, if we start from a different initial guess $\mathbf{x}_0 = (0, 1/2)$

$$\mathbf{x}_0 = \begin{pmatrix} 0 \\ 1/2 \end{pmatrix},\ \mathbf{x}_1 = \begin{pmatrix} -1 \\ -1/2 \end{pmatrix},\ \mathbf{x}_2 = \begin{pmatrix} -1.577 \\ 0.077 \end{pmatrix},\ \cdots$$

we end up in a different place.



- This is the general nature of nonlinear programming.
  - Existence - Local solutions ?
  - Uniqueness - Global solution ?

Mathematical Programming

Mathematical Program Eg.: Function Minimization via Newton

- Our example problem

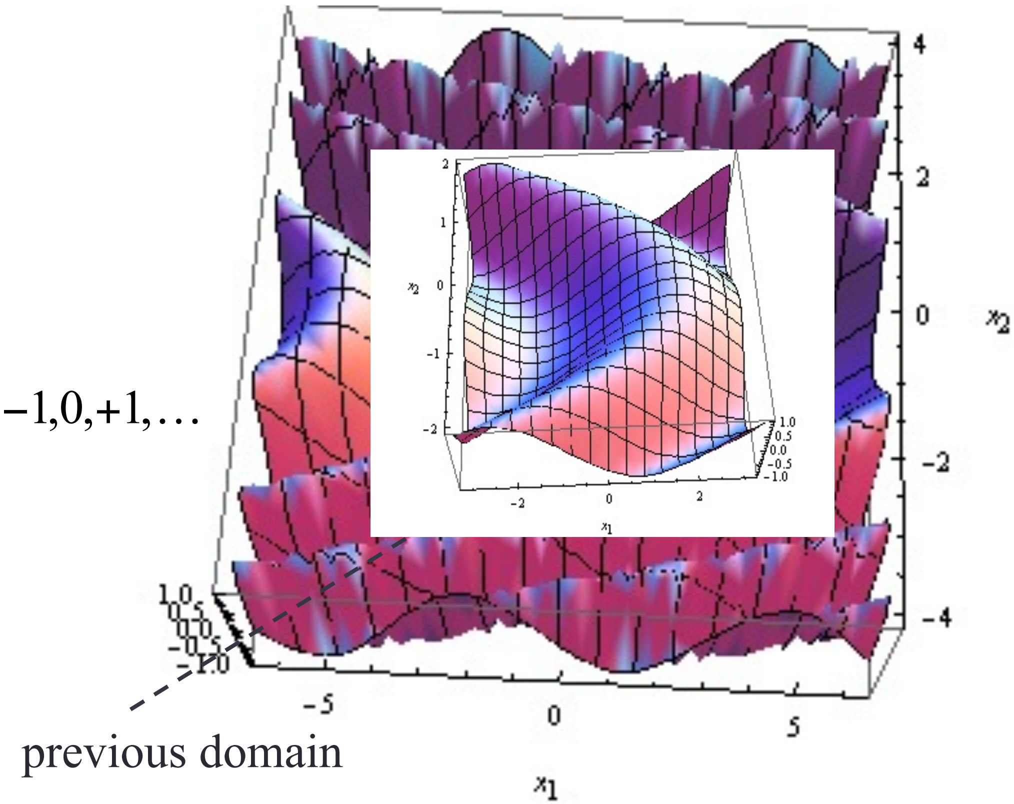
$$\min_{\mathbf{x}} J(\mathbf{x}) = \sin\left(x_1 + x_2^2\right)$$

has solutions wherever

$$x_1 + x_2^2 = \frac{2n+1}{2}\pi \qquad n = \ldots,-1,0,+1,\ldots$$

They are ubiquitous.

- This is another property nonlinear programming



Plot of $J(\mathbf{x})$ over larger domain

Mathematical Programming

Solution of Nonlinear Equations

- Many times we are faced with a problem of the form

$$
\begin{aligned}
f_1(x_1,\ldots,x_n) &= y_1 \\
&\vdots \\
f_m(x_1,\ldots,x_n) &= y_m
\end{aligned}
$$

which we abbreviate **f**(**x**) = **y** (vector notation)

- The functions f_i are nonlinear in their arguments x_i.
- For example, consider the system

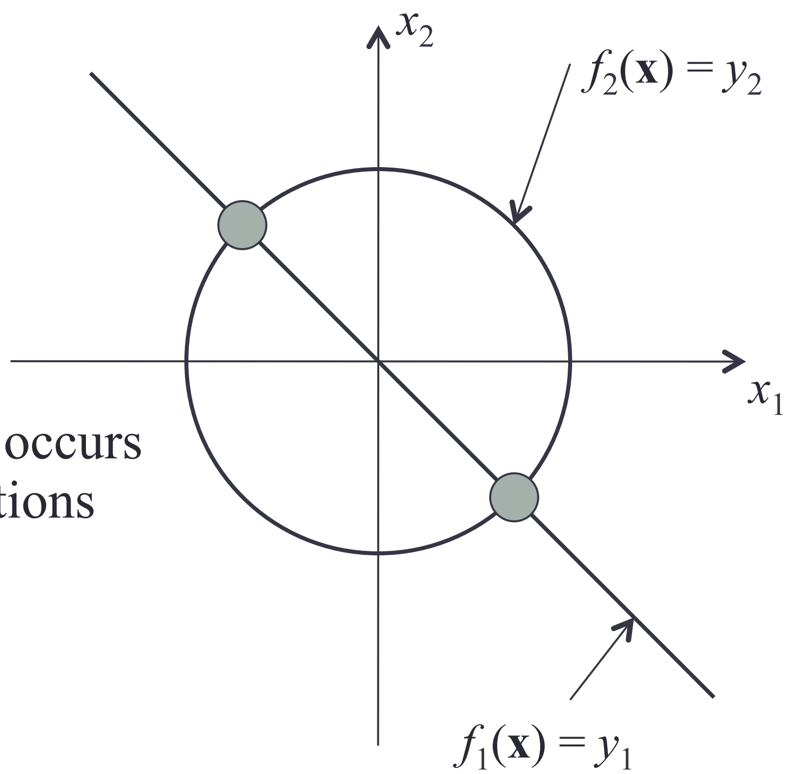
$$
\left.\begin{aligned} x_1 + x_2 &= 0 \\ x_1^2 + x_2^2 &= 1 \end{aligned}\right\} \qquad \begin{aligned} f_1(\mathbf{x}) &\equiv x_1 + x_2, \\ f_2(\mathbf{x}) &\equiv x_1^2 + x_2^2, \end{aligned} \qquad \mathbf{y} \equiv \begin{pmatrix} 0 \\ 1 \end{pmatrix}
$$

Mathematical Programming

Solution of Nonlinear Equations: Example (continued)

- Consider geometric interpretation of example problem

$$f_1(\mathbf{x}) = x_1 + x_2 = 0 = y_1,$$
$$f_2(\mathbf{x}) = x_1^2 + x_2^2 = 1 = y_2,$$



- The solution of the nonlinear problem occurs at points in the plane where both equations are satisfied.
  - Here we have two solutions

$$\mathbf{x} = \begin{pmatrix} +1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}, \begin{pmatrix} -1/\sqrt{2} \\ +1/\sqrt{2} \end{pmatrix}$$

Mathematical Programming

Solution of Nonlinear Equations: Variational Techniques

- Rather than trying to solve the nonlinear equation $\mathbf{f}(\mathbf{x}) = \mathbf{y}$ directly (there are techniques for this), another approach is to minimize the functional $J(\mathbf{x})$

$$J(\mathbf{x}) \equiv \left|\mathbf{y} - \mathbf{f}(\mathbf{x})\right|^2$$

That is,

$$\mathbf{f}(\mathbf{x}) = \mathbf{y} \quad \Rightarrow \quad \min_{\mathbf{x}} \left|\mathbf{y} - \mathbf{f}(\mathbf{x})\right|^2$$

- If we find an $\mathbf{x}_0$ such that $J(\mathbf{x}_0) = 0$, then clearly $f(\mathbf{x}_0) = \mathbf{y}$.
  - However, a minimizer $\mathbf{x}_0$ of $J(\mathbf{x})$ does not guarantee that $J(\mathbf{x}_0) = 0$ (that is, it is possible that $J(\mathbf{x}_0) > 0$ even though $J(\mathbf{x}_0)$ is a minimum)

Mathematical Programming

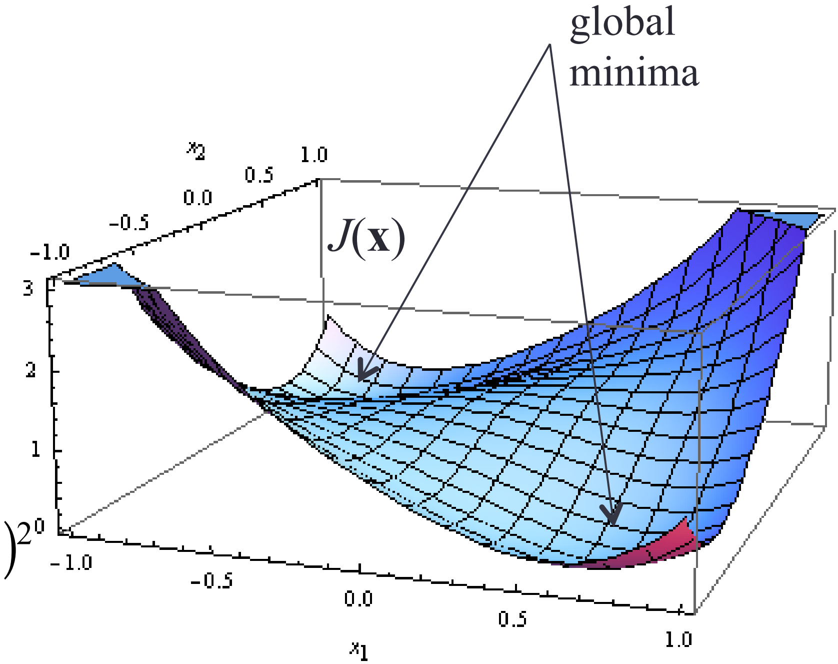
Variational Technique Example

- Recall our nonlinear problem example

$$f_1(\mathbf{x}) = x_1 + x_2 = 0 = y_1,$$
$$f_2(\mathbf{x}) = x_1^2 + x_2^2 = 1 = y_2,$$

- The variational form is

$$J(\mathbf{x}) \equiv \left|\mathbf{y} - \mathbf{f}(\mathbf{x})\right|^2 = \left(y_1 - f_1(\mathbf{x})\right)^2 + \left(y_2 - f_2(\mathbf{x})\right)^2$$
$$= \left(x_1 + x_2\right)^2 + \left(1 - x_1^2 - x_2^2\right)^2$$
$$= x_1^4 + x_2^4 + 2x_1^2 x_2^2 + 2x_1 x_2 - x_1^2 - x_2^2 + 1$$



- It has the same solutions

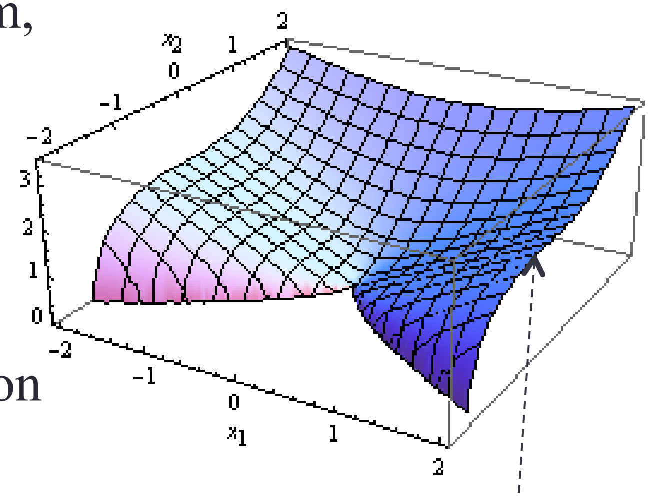
$$\mathbf{x} = \begin{pmatrix} +1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}, \begin{pmatrix} -1/\sqrt{2} \\ +1/\sqrt{2} \end{pmatrix}$$

Mathematical Programming

Constraints: A Variational Approach and Penalties

Sometimes we are faced with a constrained problem, where the solution must lie in a *feasible region* described by the equation

$$\mathbf{h}(\mathbf{x}) = \mathbf{0}$$



feasible region

- This equality indicates that the solution exists on a smooth surface (or "manifold") in $\mathbf{R}^n$
- A variation approach also works here by introducing a "tuning parameter" $c > 0$

$$\min_{\mathbf{x}} \left|\mathbf{y} - \mathbf{f}(\mathbf{x})\right|^2 + c\left|\mathbf{h}(\mathbf{x})\right|^2$$

- In general, the *penalty function* "pushes" the minimization process into the feasible region.

The magnitude of tuning parameter c determines how hard we push.

Mathematical Programming

Review

- Mathematical programming implies $\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i \mathbf{d}$
- Every mathematical programming problem has a weak (or variational) form.
- Solutions of the weak form are not guaranteed to be solutions of the original problem

Original Problem	Variational Form
$\mathbf{f}(\mathbf{x}) = \mathbf{y}$	$\min_{\mathbf{x}} \lvert \mathbf{y} - \mathbf{f}(\mathbf{x}) \rvert^2$
$\mathbf{f}(\mathbf{x}) = \mathbf{y}$ $\mathbf{h}(\mathbf{x}) = \mathbf{0}$	$\min_{\mathbf{x}} \lvert \mathbf{y} - \mathbf{f}(\mathbf{x}) \rvert^2 + c \lvert h(\mathbf{x}) \rvert^2$
$\mathbf{f}(\mathbf{x}) = \mathbf{y}$ $\mathbf{h}(\mathbf{x}) = \mathbf{0}$ $\mathbf{g}(\mathbf{x}) \le \mathbf{0}$	$\min_{\mathbf{x}} \lvert \mathbf{y} - \mathbf{f}(\mathbf{x}) \rvert^2 + c \lvert h(\mathbf{x}) \rvert^2 + d \lvert \mathbf{g}(\mathbf{x}) \rvert^2$ $d = 0$ if $\mathbf{g}(\mathbf{x}) \le \mathbf{0}$ $d > 0$ if $\mathbf{g}(\mathbf{x}) > \mathbf{0}$

(see supplemental material)

Supplemental Material

- More details on mathematical programming

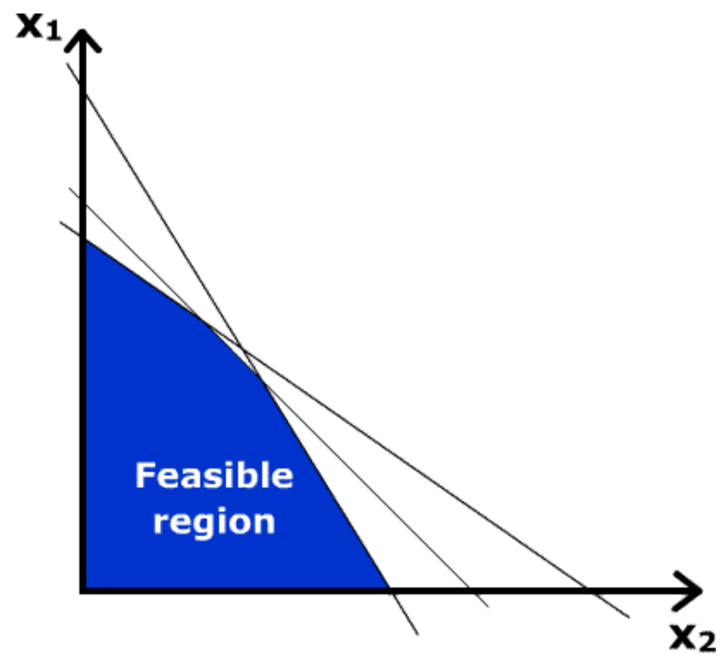
Mathematical Programming

Problems with Constraints

- Many times we are faced with problems whose solutions must remain within a specific region of parameter space
  - For example, we cannot drive magnet strengths beyond their power supply ratings.
- These constraints are usually expressed as inequalities of the form

$$\begin{aligned} g_1(x_1,\ldots,x_n) &\le 0, \\ &\vdots \\ g_m(x_1,\ldots,x_n) &\le 0, \end{aligned}$$

which can be abbreviated $\mathbf{g}(\mathbf{x}) \le 0$

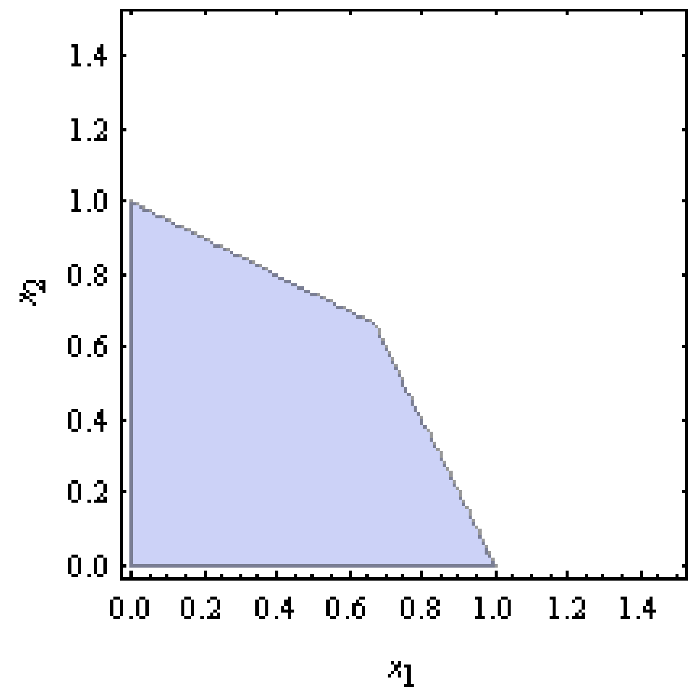


Mathematical Programming

Problems with Constraints

- The following (linear) constraints defined the shaded region in the plane:

$$
\begin{aligned}
2x_1 + x_2 - 2 &\le 0, \\
x_1 + 2x_2 - 2 &\le 0, \\
-x_1 &\le 0 \\
-x_2 &\le 0
\end{aligned}
$$



- Most nonlinear programming packages accept solution constraints if put into this form.

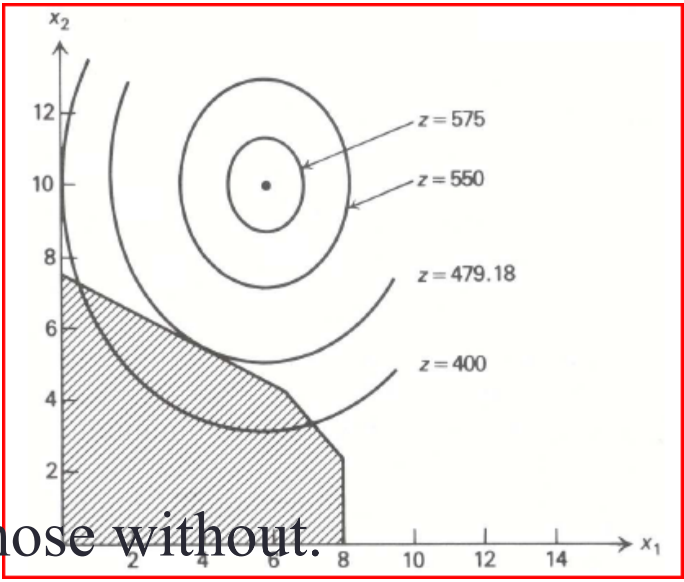
$$
\begin{gathered}
g_1(x_1, \dots, x_n) \le 0, \\
\vdots \\
g_m(x_1, \dots, x_n) \le 0,
\end{gathered}
$$

Mathematical Programming

Solution of Nonlinear Equations with Constraints

- Most nonlinear equations with constraints can be put into the vector form

$$\mathbf{f}(\mathbf{x}) = \mathbf{y}$$
$$\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$$



- In general, problems with constraints are much more difficult to solve than those without.
  - However, by using "canned" software packages and expressing the constraints in the form described, this fact is hidden from the user.

Mathematical Programming

Constrained Nonlinear Equations: Penalty Function Approach

- Starting with the nonlinear problem

$$\mathbf{f}(\mathbf{x}) = \mathbf{y}$$
$$\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$$

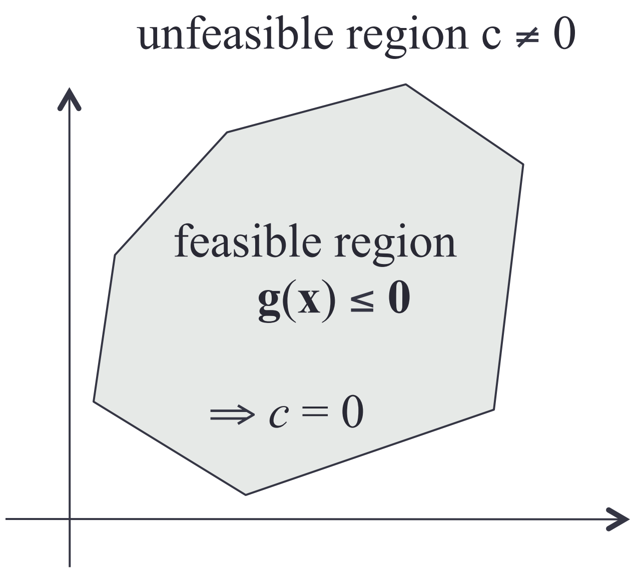
  - As before, we convert $\mathbf{f}(\mathbf{x}) = \mathbf{y}$ to the weak form

$$\min |\, \mathbf{y} - \mathbf{f}(\mathbf{x}) \,|^2$$

  - We then add a term, the "weak" form for the constraints $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$, typically called the *penalty* term

$$\min |\, \mathbf{y} - \mathbf{f}(\mathbf{x}) \,|^2 + c^{\,2} |\, \mathbf{g}(\mathbf{x}) \,|^{\,2}$$

$$c > 0 \quad \text{if } \mathbf{g}(\mathbf{x}) > 0$$
$$c = 0 \quad \text{if } \mathbf{g}(\mathbf{x}) < 0$$



Note: If $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$ then we are in the feasible region and the constraints are not binding; thus, $c = 0$.