Nonlinear Optimization: Algorithms and Models

Robert Vanderbei

June 11, 2001
McMaster University

Operations Research and Financial Engineering, Princeton University
http://www.princeton.edu/~rvdb
1 Outline

- Algorithm
  - Basic Paradigm
  - Step-Length Control
  - Diagonal Perturbation

- Convex Problems
  - Minimal Surfaces
  - Digital Audio Filters

- Nonconvex Problems
  - Celestial Mechanics
  - Putting on an Uneven Green
  - Goddard Rocket Problem
1 Outline

• Algorithm
  – Basic Paradigm
  – Step-Length Control
  – Diagonal Perturbation

• Convex Problems
  – Minimal Surfaces
  – Digital Audio Filters

• Nonconvex Problems
  – Celestial Mechanics
  – Putting on an Uneven Green
  – Goddard Rocket Problem
1 Outline

- Algorithm
  - Basic Paradigm
  - Step-Length Control
  - Diagonal Perturbation

- Convex Problems
  - Minimal Surfaces
  - Digital Audio Filters

- Nonconvex Problems
  - Celestial Mechanics
  - Putting on an Uneven Green
  - Goddard Rocket Problem
1 Outline

- Algorithm
  - Basic Paradigm
  - Step-Length Control
  - Diagonal Perturbation

- Convex Problems
  - Minimal Surfaces
  - Digital Audio Filters

- Nonconvex Problems
  - Celestial Mechanics
  - Putting on an Uneven Green
  - Goddard Rocket Problem
1 Outline

- Algorithm
  - Basic Paradigm
  - Step-Length Control
  - Diagonal Perturbation

- Convex Problems
  - Minimal Surfaces
  - Digital Audio Filters

- Nonconvex Problems
  - Celestial Mechanics
  - Putting on an Uneven Green
  - Goddard Rocket Problem
1 Outline

- Algorithm
  - Basic Paradigm
  - Step-Length Control
  - Diagonal Perturbation

- Convex Problems
  - Minimal Surfaces
  - Digital Audio Filters

- Nonconvex Problems
  - Celestial Mechanics
  - Putting on an Uneven Green
  - Goddard Rocket Problem
1 Outline

- Algorithm
  - Basic Paradigm
  - Step-Length Control
  - Diagonal Perturbation

- Convex Problems
  - Minimal Surfaces
    - Digital Audio Filters

- Nonconvex Problems
  - Celestial Mechanics
  - Putting on an Uneven Green
  - Goddard Rocket Problem
1 Outline

- Algorithm
  - Basic Paradigm
  - Step-Length Control
  - Diagonal Perturbation

- Convex Problems
  - Minimal Surfaces
  - Digital Audio Filters

- Nonconvex Problems
  - Celestial Mechanics
  - Putting on an Uneven Green
  - Goddard Rocket Problem
1 Outline

- Algorithm
  - Basic Paradigm
  - Step-Length Control
  - Diagonal Perturbation

- Convex Problems
  - Minimal Surfaces
  - Digital Audio Filters

- Nonconvex Problems
  - Celestial Mechanics
  - Putting on an Uneven Green
  - Goddard Rocket Problem
1 Outline

- Algorithm
  - Basic Paradigm
  - Step-Length Control
  - Diagonal Perturbation

- Convex Problems
  - Minimal Surfaces
  - Digital Audio Filters

- Nonconvex Problems
  - Celestial Mechanics
  - Putting on an Uneven Green
  - Goddard Rocket Problem
1 Outline

- Algorithm
  - Basic Paradigm
  - Step-Length Control
  - Diagonal Perturbation

- Convex Problems
  - Minimal Surfaces
  - Digital Audio Filters

- Nonconvex Problems
  - Celestial Mechanics
  - Putting on an Uneven Green
  - Goddard Rocket Problem
1 Outline

- Algorithm
  - Basic Paradigm
  - Step-Length Control
  - Diagonal Perturbation

- Convex Problems
  - Minimal Surfaces
  - Digital Audio Filters

- Nonconvex Problems
  - Celestial Mechanics
  - Putting on an Uneven Green
  - Goddard Rocket Problem
The Interior-Point Algorithm
2 Introduce Slack Variables

- Start with an optimization problem—for now, the simplest NLP:

  \[
  \begin{align*}
  &\text{minimize} & f(x) \\
  &\text{subject to} & h_i(x) \geq 0, \quad i = 1, \ldots, m
  \end{align*}
  \]

- Introduce slack variables to make all inequality constraints into nonnegativities:

  \[
  \begin{align*}
  &\text{minimize} & f(x) \\
  &\text{subject to} & h(x) - w = 0, \\
  & & w \geq 0
  \end{align*}
  \]
2 Introduce Slack Variables

- Start with an optimization problem—for now, the simplest NLP:

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad h_i(x) \geq 0, \quad i = 1, \ldots, m
\end{align*}
\]

- Introduce slack variables to make all inequality constraints into nonnegativities:

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad h(x) - w = 0, \\
& \quad w \geq 0
\end{align*}
\]
2 Introduce Slack Variables

- Start with an optimization problem—for now, the simplest NLP:

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad h_i(x) \geq 0, \quad i = 1, \ldots, m
\end{align*}
\]

- Introduce slack variables to make all inequality constraints into nonnegativities:

\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad h(x) - w = 0, \\
& \quad w \geq 0
\end{align*}
\]
3 Associated Log-Barrier Problem

- Replace nonnegativity constraints with logarithmic barrier terms in the objective:

  \[
  \text{minimize } f(x) - \mu \sum_{i=1}^{m} \log(w_i)
  \]

  subject to \( h(x) - w = 0 \)
3 Associated Log-Barrier Problem

- Replace nonnegativity constraints with \textit{logarithmic barrier terms} in the objective:

\[
\text{minimize} \quad f(x) - \mu \sum_{i=1}^{m} \log(w_i)
\]

\[
\text{subject to} \quad h(x) - w = 0
\]
4 First-Order Optimality Conditions

- Incorporate the equality constraints into the objective using Lagrange multipliers:

\[ L(x, w, y) = f(x) - \mu \sum_{i=1}^{m} \log(w_i) - y^T(h(x) - w) \]

- Set all derivatives to zero:

\[
\begin{align*}
\nabla f(x) - \nabla h(x)^T y &= 0 \\
-\mu W^{-1}e + y &= 0 \\
h(x) - w &= 0
\end{align*}
\]
4 First-Order Optimality Conditions

- Incorporate the equality constraints into the objective using Lagrange multipliers:

$$L(x, w, y) = f(x) - \mu \sum_{i=1}^{m} \log(w_i) - y^T(h(x) - w)$$

- Set all derivatives to zero:

$$\nabla f(x) - \nabla h(x)^T y = 0$$

$$-\mu W^{-1}e + y = 0$$

$$h(x) - w = 0$$
First-Order Optimality Conditions

- Incorporate the equality constraints into the objective using Lagrange multipliers:

\[ L(x, w, y) = f(x) - \mu \sum_{i=1}^{m} \log(w_i) - y^T(h(x) - w) \]

- Set all derivatives to zero:

\[ \nabla f(x) - \nabla h(x)^T y = 0 \]
\[ -\mu W^{-1} e + y = 0 \]
\[ h(x) - w = 0 \]
5 Symmetrize Complementarity Conditions

- Rewrite system:

\[ \nabla f(x) - \nabla h(x)^T y = 0 \]
\[ WY e = \mu e \]
\[ h(x) - w = 0 \]
5  Symmetrize Complementarity Conditions

- Rewrite system:

\[ \nabla f(x) - \nabla h(x)^T y = 0 \]

\[ WY e = \mu e \]

\[ h(x) - w = 0 \]
Apply Newton’s Method

- Apply Newton’s method to compute search directions, $\Delta x, \Delta w, \Delta y$:

$$
\begin{bmatrix}
H(x, y) & 0 & -A(x)^T \\
0 & Y & W \\
A(x) & -I & 0
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta w \\
\Delta y
\end{bmatrix}
= 
\begin{bmatrix}
-\nabla f(x) + A(x)^T y \\
\mu e - WY e \\
-h(x) + w
\end{bmatrix}.
$$

Here,

$$
H(x, y) = \nabla^2 f(x) - \sum_{i=1}^{m} y_i \nabla^2 h_i(x)
$$

and

$$
A(x) = \nabla h(x)
$$

- Note: $H(x, y)$ is positive semidefinite if $f$ is convex, each $h_i$ is concave, and each $y_i \geq 0$. 

6  Apply Newton’s Method

- Apply Newton’s method to compute search directions, $\Delta x$, $\Delta w$, $\Delta y$:

\[
\begin{bmatrix}
H(x, y) & 0 & -A(x)^T \\
0 & Y & W \\
A(x) & -I & 0
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta w \\
\Delta y
\end{bmatrix}
= 
\begin{bmatrix}
-\nabla f(x) + A(x)^T y \\
\mu e - W Y e \\
-h(x) + w
\end{bmatrix}.
\]

Here,

\[
H(x, y) = \nabla^2 f(x) - \sum_{i=1}^{m} y_i \nabla^2 h_i(x)
\]

and

\[
A(x) = \nabla h(x)
\]

- Note: $H(x, y)$ is positive semidefinite if $f$ is convex, each $h_i$ is concave, and each $y_i \geq 0$. 
Apply Newton’s Method

Apply Newton’s method to compute search directions, $\Delta x$, $\Delta w$, $\Delta y$:

\[
\begin{bmatrix}
H(x, y) & 0 & -A(x)^T \\
0 & Y & W \\
A(x) & -I & 0
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta w \\
\Delta y
\end{bmatrix}
=
\begin{bmatrix}
-\nabla f(x) + A(x)^T y \\
\mu e - WY e \\
-h(x) + w
\end{bmatrix}.
\]

Here,

\[ H(x, y) = \nabla^2 f(x) - \sum_{i=1}^{m} y_i \nabla^2 h_i(x) \]

and

\[ A(x) = \nabla h(x) \]

Note: $H(x, y)$ is positive semidefinite if $f$ is convex, each $h_i$ is concave, and each $y_i \geq 0$. 
7 Reduced KKT System

- Use second equation to solve for $\Delta w$. Result is the reduced KKT system:

$$
\begin{bmatrix}
-H(x, y) & A^T(x) \\
A(x) & WY^{-1}
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta y
\end{bmatrix}
=
\begin{bmatrix}
\nabla f(x) - A^T(x)y \\
-h(x) + \mu Y^{-1}e
\end{bmatrix}
$$

- Iterate:

$$
x^{(k+1)} = x^{(k)} + \alpha^{(k)} \Delta x^{(k)}
$$

$$
w^{(k+1)} = w^{(k)} + \alpha^{(k)} \Delta w^{(k)}
$$

$$
y^{(k+1)} = y^{(k)} + \alpha^{(k)} \Delta y^{(k)}
$$
7 Reduced KKT System

- Use second equation to solve for $\Delta w$. Result is the reduced KKT system:

$$
\begin{bmatrix}
-H(x, y) & A^T(x) \\
A(x) & WY^{-1}
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta y
\end{bmatrix}
=
\begin{bmatrix}
\nabla f(x) - A^T(x)y \\
h(x) + \mu Y^{-1}e
\end{bmatrix}
$$

- Iterate:

$$
x^{(k+1)} = x^{(k)} + \alpha^{(k)} \Delta x^{(k)} \\
w^{(k+1)} = w^{(k)} + \alpha^{(k)} \Delta w^{(k)} \\
y^{(k+1)} = y^{(k)} + \alpha^{(k)} \Delta y^{(k)}
$$
Reduced KKT System

Use second equation to solve for $\Delta w$. Result is the reduced KKT system:

$$
\begin{bmatrix}
-H(x, y) & A^T(x) \\
A(x) & WY^{-1}
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta y
\end{bmatrix}
= 
\begin{bmatrix}
\nabla f(x) - A^T(x)y \\
-h(x) + \mu Y^{-1}e
\end{bmatrix}
$$

Iterate:

$$
x^{(k+1)} = x^{(k)} + \alpha^{(k)} \Delta x^{(k)}
$$

$$
w^{(k+1)} = w^{(k)} + \alpha^{(k)} \Delta w^{(k)}
$$

$$
y^{(k+1)} = y^{(k)} + \alpha^{(k)} \Delta y^{(k)}
$$
8 Convex vs. Nonconvex Optimization Probs

Nonlinear Programming (NLP)

minimize \( f(x) \)
subject to \( h_i(x) = 0, \quad i \in \mathcal{E}, \)
\( h_i(x) \geq 0, \quad i \in \mathcal{I}. \)

NLP is convex if

- \( h_i \)'s in equality constraints are affine;
- \( h_i \)'s in inequality constraints are concave;
- \( f \) is convex;

NLP is smooth if

- All are twice continuously differentiable.
Nonlinear Programming (NLP)

$$\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad h_i(x) = 0, \quad i \in E, \\
& \quad h_i(x) \geq 0, \quad i \in I.
\end{align*}$$

NLP is **convex** if

- $h_i$’s in equality constraints are affine;
- $h_i$’s in inequality constraints are concave;
- $f$ is convex;

NLP is **smooth** if

- All are twice continuously differentiable.
Nonlinear Programming (NLP)

\[
\begin{aligned}
\text{minimize} & \quad f(x) \\
\text{subject to} & \quad h_i(x) = 0, \quad i \in \mathcal{E}, \\
& \quad h_i(x) \geq 0, \quad i \in \mathcal{I}.
\end{aligned}
\]

NLP is \textit{convex} if

- \(h_i\)'s in equality constraints are affine;
- \(h_i\)'s in inequality constraints are concave;
- \(f\) is convex;

NLP is \textit{smooth} if

- All are twice continuously differentiable.
9 Modifications for Convex Optimization

For convex nonquadratic optimization, it does not suffice to choose the steplength $\alpha$ simply to maintain positivity of nonnegative variables.

- Consider, e.g., minimizing
  \[
  f(x) = (1 + x^2)^{1/2}.
  \]

- The iterates can be computed explicitly:
  \[
  x^{(k+1)} = -(x^{(k)})^3
  \]

- Converges if and only if $|x| \leq 1$.

- Reason: away from 0, function is too linear.
Modifications for Convex Optimization

For convex nonquadratic optimization, it does not suffice to choose the steplength $\alpha$ simply to maintain positivity of nonnegative variables.

- Consider, e.g., minimizing

  $$ f(x) = (1 + x^2)^{1/2}. $$

- The iterates can be computed explicitly:

  $$ x^{(k+1)} = -(x^{(k)})^3 $$

- Converges if and only if $|x| \leq 1$.

- Reason: away from 0, function is too linear.
For convex nonquadratic optimization, it does not suffice to choose the steplength $\alpha$ simply to maintain positivity of nonnegative variables.

- Consider, e.g., minimizing

$$f(x) = (1 + x^2)^{1/2}.$$ 

- The iterates can be computed explicitly:

$$x^{(k+1)} = -(x^{(k)})^3$$

- Converges if and only if $|x| \leq 1$.

- Reason: away from 0, function is too linear.
## 9 Modifications for Convex Optimization

For convex nonquadratic optimization, it does not suffice to choose the steplength $\alpha$ simply to maintain positivity of nonnegative variables.

- Consider, e.g., minimizing

  $$f(x) = (1 + x^2)^{1/2}.$$  

- The iterates can be computed explicitly:

  $$x^{(k+1)} = -(x^{(k)})^3$$

- Converges if and only if $|x| \leq 1$.

- Reason: away from 0, function is too linear.
For convex nonquadratic optimization, it does not suffice to choose the steplength $\alpha$ simply to maintain positivity of nonnegative variables.

- Consider, e.g., minimizing

$$f(x) = (1 + x^2)^{1/2}.$$ 

- The iterates can be computed explicitly:

$$x^{(k+1)} = - (x^{(k)})^3.$$ 

- Converges if and only if $|x| \leq 1$.

- Reason: away from 0, function is too linear.
10 Step-Length Control

A filter-type method is used to guide the choice of step length $\alpha$.

Define the dual normal matrix:

$$N(x, y, w) = H(x, y) + A^T(x)W^{-1}YA(x).$$

**Theorem 1** Suppose that $N(x, y, w)$ is positive definite.

1. *If current solution is primal infeasible, then $(\Delta x, \Delta w)$ is a descent direction for the infeasibility $\|h(x) - w\|$.*

2. *If current solution is primal feasible, then $(\Delta x, \Delta w)$ is a descent direction for the barrier function.*

Shorten $\alpha$ until $(\Delta x, \Delta w)$ is a descent direction for either the infeasibility or the barrier function.
10 Step-Length Control

A filter-type method is used to guide the choice of steplength $\alpha$.

Define the dual normal matrix:

$$N(x, y, w) = H(x, y) + A^T(x)W^{-1}YA(x).$$

**Theorem 1** Suppose that $N(x, y, w)$ is positive definite.

1. If current solution is primal infeasible, then $(\Delta x, \Delta w)$ is a descent direction for the infeasibility $\|h(x) - w\|$.

2. If current solution is primal feasible, then $(\Delta x, \Delta w)$ is a descent direction for the barrier function.

Shorten $\alpha$ until $(\Delta x, \Delta w)$ is a descent direction for either the infeasibility or the barrier function.
10 Step-Length Control

A filter-type method is used to guide the choice of step length $\alpha$.

Define the dual normal matrix:

$$N(x, y, w) = H(x, y) + A^T(x)W^{-1}YA(x).$$

**Theorem 1** Suppose that $N(x, y, w)$ is positive definite.

1. If current solution is primal infeasible, then $(\Delta x, \Delta w)$ is a descent direction for the infeasibility $\|h(x) - w\|$.

2. If current solution is primal feasible, then $(\Delta x, \Delta w)$ is a descent direction for the barrier function.

Shorten $\alpha$ until $(\Delta x, \Delta w)$ is a descent direction for either the infeasibility or the barrier function.
10 Step-Length Control

A filter-type method is used to guide the choice of steplength $\alpha$.

Define the dual normal matrix:

$$N(x, y, w) = H(x, y) + A^T(x)W^{-1}YA(x).$$

**Theorem 1** Suppose that $N(x, y, w)$ is positive definite.

1. If current solution is primal infeasible, then $(\Delta x, \Delta w)$ is a descent direction for the infeasibility $\|h(x) - w\|$.
2. If current solution is primal feasible, then $(\Delta x, \Delta w)$ is a descent direction for the barrier function.

Shorten $\alpha$ until $(\Delta x, \Delta w)$ is a descent direction for either the infeasibility or the barrier function.
10 Step-Length Control

A filter-type method is used to guide the choice of steplength $\alpha$.

Define the dual normal matrix:

$$N(x, y, w) = H(x, y) + A^T(x)W^{-1}YA(x).$$

**Theorem 1** Suppose that $N(x, y, w)$ is positive definite.

1. If current solution is primal infeasible, then $(\Delta x, \Delta w)$ is a descent direction for the infeasibility $\|h(x) - w\|$.

2. If current solution is primal feasible, then $(\Delta x, \Delta w)$ is a descent direction for the barrier function.

Shorten $\alpha$ until $(\Delta x, \Delta w)$ is a descent direction for either the infeasibility or the barrier function.
10  Step-Length Control

A filter-type method is used to guide the choice of steplength $\alpha$.

Define the dual normal matrix:

$$N(x, y, w) = H(x, y) + A^T(x)W^{-1}YA(x).$$

**Theorem 1**  Suppose that $N(x, y, w)$ is positive definite.

1. If current solution is primal infeasible, then $(\Delta x, \Delta w)$ is a descent direction for the infeasibility $\|h(x) - w\|$.

2. If current solution is primal feasible, then $(\Delta x, \Delta w)$ is a descent direction for the barrier function.

Shorten $\alpha$ until $(\Delta x, \Delta w)$ is a descent direction for either the infeasibility or the barrier function.
10 Step-Length Control

A filter-type method is used to guide the choice of steplength $\alpha$.

Define the dual normal matrix:

$$N(x, y, w) = H(x, y) + A^T(x)W^{-1}YA(x).$$

**Theorem 1** Suppose that $N(x, y, w)$ is positive definite.

1. If current solution is primal infeasible, then $(\Delta x, \Delta w)$ is a descent direction for the infeasibility $\|h(x) - w\|$.

2. If current solution is primal feasible, then $(\Delta x, \Delta w)$ is a descent direction for the barrier function.

Shorten $\alpha$ until $(\Delta x, \Delta w)$ is a descent direction for either the infeasibility or the barrier function.
10 Step-Length Control

A filter-type method is used to guide the choice of steplength $\alpha$.

Define the dual normal matrix:

$$N(x, y, w) = H(x, y) + A^T(x)W^{-1}YA(x).$$

**Theorem 1** Suppose that $N(x, y, w)$ is positive definite.

1. If current solution is primal infeasible, then $(\Delta x, \Delta w)$ is a descent direction for the infeasibility $\|h(x) - w\|$.

2. If current solution is primal feasible, then $(\Delta x, \Delta w)$ is a descent direction for the barrier function.

Shorten $\alpha$ until $(\Delta x, \Delta w)$ is a descent direction for either the infeasibility or the barrier function.
11 Nonconvex Optimization: Diagonal Perturbation

- If $H(x, y)$ is not positive semidefinite then $N(x, y, w)$ might fail to be positive definite.
- In such a case, we lose the descent properties given in previous theorem.
- To regain those properties, we perturb the Hessian: $\tilde{H}(x, y) = H(x, y) + \lambda I$.
- And compute search directions using $\tilde{H}$ instead of $H$.

Notation: let $\tilde{N}$ denote the dual normal matrix associated with $\tilde{H}$.

**Theorem 2** If $\tilde{N}$ is positive definite, then $(\Delta x, \Delta w, \Delta y)$ is a descent direction for

1. the primal infeasibility, $\|h(x) - w\|$;
2. the noncomplementarity, $w^T y$. 
If $H(x, y)$ is not positive semidefinite then $N(x, y, w)$ might fail to be positive definite.

- In such a case, we lose the descent properties given in previous theorem.
- To regain those properties, we perturb the Hessian: $\tilde{H}(x, y) = H(x, y) + \lambda I$.
- And compute search directions using $\tilde{H}$ instead of $H$.

Notation: let $\tilde{N}$ denote the dual normal matrix associated with $\tilde{H}$.

**Theorem 2** If $\tilde{N}$ is positive definite, then $(\Delta x, \Delta w, \Delta y)$ is a descent direction for

1. the primal infeasibility, $\|h(x) - w\|$;
2. the noncomplementarity, $w^T y$. 

Nonconvex Optimization: Diagonal Perturbation

- If $H(x, y)$ is not positive semidefinite then $N(x, y, w)$ might fail to be positive definite.
- In such a case, we lose the descent properties given in previous theorem.
- To regain those properties, we perturb the Hessian: $\tilde{H}(x, y) = H(x, y) + \lambda I$.
- And compute search directions using $\tilde{H}$ instead of $H$.

Notation: let $\tilde{N}$ denote the dual normal matrix associated with $\tilde{H}$.

**Theorem 2** If $\tilde{N}$ is positive definite, then $(\Delta x, \Delta w, \Delta y)$ is a descent direction for

1. the primal infeasibility, $\|h(x) - w\|$;
2. the noncomplementarity, $w^T y$. 
If $H(x, y)$ is not positive semidefinite then $N(x, y, w)$ might fail to be positive definite. In such a case, we lose the descent properties given in previous theorem. To regain those properties, we perturb the Hessian: $\tilde{H}(x, y) = H(x, y) + \lambda I$. And compute search directions using $\tilde{H}$ instead of $H$.

Notation: let $\tilde{N}$ denote the dual normal matrix associated with $\tilde{H}$.

**Theorem 2** If $\tilde{N}$ is positive definite, then $(\Delta x, \Delta w, \Delta y)$ is a descent direction for

1. the primal infeasibility, $\|h(x) - w\|$;
2. the noncomplementarity, $w^T y$. 
11 Nonconvex Optimization: Diagonal Perturbation

- If $H(x, y)$ is not positive semidefinite then $N(x, y, w)$ might fail to be positive definite.
- In such a case, we lose the descent properties given in previous theorem.
- To regain those properties, we perturb the Hessian: $\tilde{H}(x, y) = H(x, y) + \lambda I$.
- And compute search directions using $\tilde{H}$ instead of $H$.

Notation: let $\tilde{N}$ denote the dual normal matrix associated with $\tilde{H}$.

**Theorem 2** If $\tilde{N}$ is positive definite, then $(\Delta x, \Delta w, \Delta y)$ is a descent direction for

1. the primal infeasibility, $\| h(x) - w \|$;
2. the noncomplementarity, $w^T y$. 


11 Nonconvex Optimization: Diagonal Perturbation

- If $H(x, y)$ is not positive semidefinite then $N(x, y, w)$ might fail to be positive definite.
- In such a case, we lose the descent properties given in previous theorem.
- To regain those properties, we perturb the Hessian: $\tilde{H}(x, y) = H(x, y) + \lambda I$.
- And compute search directions using $\tilde{H}$ instead of $H$.

Notation: let $\tilde{N}$ denote the dual normal matrix associated with $\tilde{H}$.

**Theorem 2** If $\tilde{N}$ is positive definite, then $(\Delta x, \Delta w, \Delta y)$ is a descent direction for

1. the primal infeasibility, $\|h(x) - w\|$;
2. the noncomplementarity, $w^T y$. 
If $H(x, y)$ is not positive semidefinite then $N(x, y, w)$ might fail to be positive definite.

In such a case, we lose the descent properties given in previous theorem.

To regain those properties, we perturb the Hessian: $\tilde{H}(x, y) = H(x, y) + \lambda I$.

And compute search directions using $\tilde{H}$ instead of $H$.

Notation: let $\tilde{N}$ denote the dual normal matrix associated with $\tilde{H}$.

**Theorem 2** If $\tilde{N}$ is positive definite, then $(\Delta x, \Delta w, \Delta y)$ is a descent direction for

1. the primal infeasibility, $\|h(x) - w\|$;
2. the noncomplementarity, $w^T y$. 
Nonconvex Optimization: Diagonal Perturbation

- If $H(x, y)$ is not positive semidefinite then $N(x, y, w)$ might fail to be positive definite.
- In such a case, we lose the descent properties given in previous theorem.
- To regain those properties, we perturb the Hessian: $\tilde{H}(x, y) = H(x, y) + \lambda I$.
- And compute search directions using $\tilde{H}$ instead of $H$.

Notation: let $\tilde{N}$ denote the dual normal matrix associated with $\tilde{H}$.

**Theorem 2** If $\tilde{N}$ is positive definite, then $(\Delta x, \Delta w, \Delta y)$ is a descent direction for

1. the primal infeasibility, $\|h(x) - w\|$;
2. the noncomplementarity, $w^T y$.
Nonconvex Optimization: Diagonal Perturbation

- If $H(x, y)$ is not positive semidefinite then $N(x, y, w)$ might fail to be positive definite.
- In such a case, we lose the descent properties given in previous theorem.
- To regain those properties, we perturb the Hessian: $\tilde{H}(x, y) = H(x, y) + \lambda I$.
- And compute search directions using $\tilde{H}$ instead of $H$.

Notation: let $\tilde{N}$ denote the dual normal matrix associated with $\tilde{H}$.

Theorem 2  If $\tilde{N}$ is positive definite, then $(\Delta x, \Delta w, \Delta y)$ is a descent direction for

1. the primal infeasibility, $\|h(x) - w\|$;

2. the noncomplementarity, $w^T y$. 
If \( H(x, y) \) is not positive semidefinite then \( N(x, y, w) \) might fail to be positive definite.

In such a case, we lose the descent properties given in previous theorem.

To regain those properties, we perturb the Hessian: \( \tilde{H}(x, y) = H(x, y) + \lambda I \).

And compute search directions using \( \tilde{H} \) instead of \( H \).

Notation: let \( \tilde{N} \) denote the dual normal matrix associated with \( \tilde{H} \).

**Theorem 2** If \( \tilde{N} \) is positive definite, then \( (\Delta x, \Delta w, \Delta y) \) is a descent direction for

1. the primal infeasibility, \( ||h(x) - w|| \);
2. the noncomplementarity, \( w^T y \).
Notes:

- Not necessarily a descent direction for dual infeasibility.

- A line search is performed to find a value of $\lambda$ within a factor of 2 of the smallest permissible value.
Notes:

- Not necessarily a descent direction for dual infeasibility.

- A line search is performed to find a value of $\lambda$ within a factor of 2 of the smallest permissible value.
• Not necessarily a descent direction for dual infeasibility.

• A line search is performed to find a value of $\lambda$ within a factor of 2 of the smallest permissible value.
Theorem 3  If the problem is convex and and the current solution is not optimal and ..., then for any slack variable, say $w_i$, we have $w_i = 0$ implies $\Delta w_i \geq 0$.

- To paraphrase: for convex problems, as slack variables get small they tend to get large again. This is an antijamming theorem.
- A recent example of Wächter and Biegler shows that for nonconvex problems, jamming really can occur.
- Recent modification:
  - if a slack variable gets small and
  - its component of the step direction contributes to making a very short step,
  - then increase this slack variable to the average size of the variables the “mainstream” slack variables.
- This modification corrects all examples of jamming that we know about.
Theorem 3  *If the problem is convex and and the current solution is not optimal and ..., then for any slack variable, say* $w_i$, *we have* $w_i = 0$ *implies* $\Delta w_i \geq 0$.

- To paraphrase: for convex problems, as slack variables get small they tend to get large again. This is an antijamming theorem.
- A recent example of Wächter and Biegler shows that for nonconvex problems, jamming really can occur.
- Recent modification:
  - if a slack variable gets small and
  - its component of the step direction contributes to making a very short step,
  - then increase this slack variable to the average size of the variables the “mainstream” slack variables.
- This modification corrects all examples of jamming that we know about.
Theorem 3  *If the problem is convex and and the current solution is not optimal and ..., then for any slack variable, say \( w_i \), we have \( w_i = 0 \) implies \( \Delta w_i \geq 0 \).*

- To paraphrase: for convex problems, as slack variables get small they tend to get large again. This is an antijamming theorem.
- A recent example of Wächter and Biegler shows that for nonconvex problems, jamming really can occur.
- Recent modification:
  - if a slack variable gets small and
  - its component of the step direction contributes to making a very short step,
  - then increase this slack variable to the average size of the variables the “mainstream” slack variables.
- This modification corrects all examples of jamming that we know about.
Theorem 3 If the problem is convex and the current solution is not optimal and ..., then for any slack variable, say \( w_i \), we have \( w_i = 0 \) implies \( \Delta w_i \geq 0 \).

- To paraphrase: for convex problems, as slack variables get small they tend to get large again. This is an antijamming theorem.
- A recent example of Wächter and Biegler shows that for nonconvex problems, jamming really can occur.
- Recent modification:
  - if a slack variable gets small and
  - its component of the step direction contributes to making a very short step,
  - then increase this slack variable to the average size of the variables the “mainstream” slack variables.
- This modification corrects all examples of jamming that we know about.
Theorem 3: If the problem is convex and the current solution is not optimal and ..., then for any slack variable, say $w_i$, we have $w_i = 0$ implies $\Delta w_i \geq 0$.

- To paraphrase: for convex problems, as slack variables get small they tend to get large again. This is an antijamming theorem.
- A recent example of Wächter and Biegler shows that for nonconvex problems, jamming really can occur.
- Recent modification:
  - if a slack variable gets small and
  - its component of the step direction contributes to making a very short step,
  - then increase this slack variable to the average size of the variables the "mainstream" slack variables.
- This modification corrects all examples of jamming that we know about.
Theorem 3  If the problem is convex and and the current solution is not optimal and ..., then for any slack variable, say $w_i$, we have $w_i = 0$ implies $\Delta w_i \geq 0$.

- To paraphrase: for convex problems, as slack variables get small they tend to get large again. This is an antijamming theorem.

- A recent example of Wächter and Biegler shows that for nonconvex problems, jamming really can occur.

- Recent modification:
  - if a slack variable gets small and
  - its component of the step direction contributes to making a very short step,
  - then increase this slack variable to the average size of the variables the “mainstream” slack variables.

- This modification corrects all examples of jamming that we know about.
13  Nonconvex Optimization: Jamming

Theorem 3  If the problem is convex and the current solution is not optimal and ..., then for any slack variable, say $w_i$, we have $w_i = 0$ implies $\Delta w_i \geq 0$.

- To paraphrase: for convex problems, as slack variables get small they tend to get large again. This is an antijamming theorem.

- A recent example of Wächter and Biegler shows that for nonconvex problems, jamming really can occur.

- Recent modification:
  - if a slack variable gets small and
  - its component of the step direction contributes to making a very short step,
  - then increase this slack variable to the average size of the variables the “mainstream” slack variables.

- This modification corrects all examples of jamming that we know about.
Theorem 3  If the problem is convex and and the current solution is not optimal and ..., then for any slack variable, say $w_i$, we have $w_i = 0$ implies $\Delta w_i \geq 0$.

- To paraphrase: for convex problems, as slack variables get small they tend to get large again. This is an antijamming theorem.
- A recent example of Wächter and Biegler shows that for nonconvex problems, jamming really can occur.
- Recent modification:
  - if a slack variable gets small and
  - its component of the step direction contributes to making a very short step,
  - then increase this slack variable to the average size of the variables the “mainstream” slack variables.
- This modification corrects all examples of jamming that we know about.
Theorem 3  If the problem is convex and the current solution is not optimal and ..., then for any slack variable, say $w_i$, we have $w_i = 0$ implies $\Delta w_i \geq 0$.

- To paraphrase: for convex problems, as slack variables get small they tend to get large again. This is an antijamming theorem.
- A recent example of Wächter and Biegler shows that for nonconvex problems, jamming really can occur.
- Recent modification:
  - if a slack variable gets small and
  - its component of the step direction contributes to making a very short step,
  - then increase this slack variable to the average size of the variables the “mainstream” slack variables.
- This modification corrects all examples of jamming that we know about.
Modifications for General Problem Formulations

- Bounds, ranges, and free variables are all treated implicitly as described in Linear Programming: Foundations and Extensions (LP:F&E).

- Net result is following reduced KKT system:

\[
\begin{bmatrix}
-(H(x, y) + D) & A^T(x) \\
A(x) & E
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta y
\end{bmatrix}
= 
\begin{bmatrix}
\Phi_1 \\
\Phi_2
\end{bmatrix}
\]

- Here, \( D \) and \( E \) are positive definite diagonal matrices.

- Note that \( D \) helps reduce frequency of diagonal perturbation.

- Choice of barrier parameter \( \mu \) and initial solution, if none is provided, is described in the paper.

- Stopping rules, matrix reordering heuristics, etc. are as described in LP:F&E.
Modifications for General Problem Formulations

• Bounds, ranges, and free variables are all treated implicitly as described in Linear Programming: Foundations and Extensions (LP:F&E).

• Net result is following reduced KKT system:

\[
\begin{bmatrix}
-(H(x, y) + D) & A^T(x) \\
A(x) & E
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta y
\end{bmatrix} =
\begin{bmatrix}
\Phi_1 \\
\Phi_2
\end{bmatrix}
\]

• Here, \(D\) and \(E\) are positive definite diagonal matrices.

• Note that \(D\) helps reduce frequency of diagonal perturbation.

• Choice of barrier parameter \(\mu\) and initial solution, if none is provided, is described in the paper.

• Stopping rules, matrix reordering heuristics, etc. are as described in LP:F&E.
Modifications for General Problem Formulations

- Bounds, ranges, and free variables are all treated implicitly as described in *Linear Programming: Foundations and Extensions (LP:F&E)*.
- Net result is following reduced KKT system:

\[
\begin{bmatrix}
-(H(x, y) + D) & A^T(x) \\
A(x) & E
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta y
\end{bmatrix}
= \begin{bmatrix}
\Phi_1 \\
\Phi_2
\end{bmatrix}
\]

- Here, \( D \) and \( E \) are positive definite diagonal matrices.
- Note that \( D \) helps reduce frequency of diagonal perturbation.
- Choice of barrier parameter \( \mu \) and initial solution, if none is provided, is described in the paper.
- Stopping rules, matrix reordering heuristics, etc. are as described in *LP:F&E*. 
Modifications for General Problem Formulations

- Bounds, ranges, and free variables are all treated implicitly as described in Linear Programming: Foundations and Extensions (LP:F&E).

- Net result is following reduced KKT system:

$$
\begin{pmatrix}
-(H(x, y) + D) & A^T(x) \\
A(x) & E
\end{pmatrix}
\begin{pmatrix}
\Delta x \\
\Delta y
\end{pmatrix}
=
\begin{pmatrix}
\Phi_1 \\
\Phi_2
\end{pmatrix}
$$

- Here, $D$ and $E$ are positive definite diagonal matrices.

- Note that $D$ helps reduce frequency of diagonal perturbation.

- Choice of barrier parameter $\mu$ and initial solution, if none is provided, is described in the paper.

- Stopping rules, matrix reordering heuristics, etc. are as described in LP:F&E.
Modifications for General Problem Formulations

- Bounds, ranges, and free variables are all treated implicitly as described in Linear Programming: Foundations and Extensions (LP:F&E).

- Net result is following reduced KKT system:

\[
\begin{bmatrix}
    -(H(x, y) + D) & A^T(x) \\
    A(x) & E
\end{bmatrix}
\begin{bmatrix}
    \Delta x \\
    \Delta y
\end{bmatrix}
= \begin{bmatrix}
    \Phi_1 \\
    \Phi_2
\end{bmatrix}
\]

- Here, \( D \) and \( E \) are positive definite diagonal matrices.

- Note that \( D \) helps reduce frequency of diagonal perturbation.

- Choice of barrier parameter \( \mu \) and initial solution, if none is provided, is described in the paper.

- Stopping rules, matrix reordering heuristics, etc. are as described in LP:F&E.
14 Modifications for General Problem Formulations

- Bounds, ranges, and free variables are all treated implicitly as described in *Linear Programming: Foundations and Extensions (LP:F&E)*.

- Net result is following reduced KKT system:

\[
\begin{bmatrix}
-(H(x, y) + D) & A^T(x) \\
A(x) & E
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta y
\end{bmatrix}
= \begin{bmatrix}
\Phi_1 \\
\Phi_2
\end{bmatrix}
\]

- Here, \( D \) and \( E \) are positive definite diagonal matrices.

- Note that \( D \) helps reduce frequency of diagonal perturbation.

- Choice of barrier parameter \( \mu \) and initial solution, if none is provided, is described in the paper.

- Stopping rules, matrix reordering heuristics, etc. are as described in *LP:F&E*. 

14 Modifications for General Problem Formulations

- Bounds, ranges, and free variables are all treated implicitly as described in Linear Programming: Foundations and Extensions (LP:F&E).

- Net result is following reduced KKT system:

\[
\begin{bmatrix}
-(H(x, y) + D) & A^T(x) \\
A(x) & E
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta y
\end{bmatrix}
= \begin{bmatrix}
\Phi_1 \\
\Phi_2
\end{bmatrix}
\]

- Here, \( D \) and \( E \) are positive definite diagonal matrices.

- Note that \( D \) helps reduce frequency of diagonal perturbation.

- Choice of barrier parameter \( \mu \) and initial solution, if none is provided, is described in the paper.

- Stopping rules, matrix reordering heuristics, etc. are as described in LP:F&E.
Examples: Convex Optimization Models
Minimal Surfaces

- Given: a domain $D$ in $\mathbb{R}^2$ and an embedding $x = (x_1, x_2, x_3)$ of its boundary $\partial D$ in $\mathbb{R}^3$;
- Find: an embedding of the entire domain into $\mathbb{R}^3$ that is consistent with the boundary embedding and has minimal surface area:

$$\minimize \iint_D \left\| \frac{\partial x}{\partial s} \times \frac{\partial x}{\partial t} \right\| \, ds \, dt$$

subject to $x(s, t)$ fixed for $(s, t) \in \partial D$
$x_1(s, t)$ fixed for $(s, t) \in D$
$x_2(s, t)$ fixed for $(s, t) \in D$

The specific problems coded below take $D$ to be either a square or an annulus.
15 Minimal Surfaces

- Given: a domain \( D \) in \( \mathbb{R}^2 \) and an embedding \( x = (x_1, x_2, x_3) \) of its boundary \( \partial D \) in \( \mathbb{R}^3 \);
- Find: an embedding of the entire domain into \( \mathbb{R}^3 \) that is consistent with the boundary embedding and has minimal surface area:

\[
\minimize \int \int_D \left\| \frac{\partial x}{\partial s} \times \frac{\partial x}{\partial t} \right\| \, ds \, dt
\]

subject to
- \( x(s, t) \) fixed for \( (s, t) \in \partial D \)
- \( x_1(s, t) \) fixed for \( (s, t) \in D \)
- \( x_2(s, t) \) fixed for \( (s, t) \in D \)

The specific problems coded below take \( D \) to be either a square or an annulus.
15 Minimal Surfaces

- Given: a domain $D$ in $\mathbb{R}^2$ and an embedding $x = (x_1, x_2, x_3)$ of its boundary $\partial D$ in $\mathbb{R}^3$;
- Find: an embedding of the entire domain into $\mathbb{R}^3$ that is consistent with the boundary embedding and has minimal surface area:

$$\text{minimize} \quad \iint_D \left\| \frac{\partial x}{\partial s} \times \frac{\partial x}{\partial t} \right\| \, ds \, dt$$

subject to $x(s, t)$ fixed for $(s, t) \in \partial D$

$x_1(s, t)$ fixed for $(s, t) \in D$

$x_2(s, t)$ fixed for $(s, t) \in D$

The specific problems coded below take $D$ to be either a square or an annulus.
16 Specific Example

Scherk.mod with $D$ discretized into a $64 \times 64$ grid gives the following results:

- Constraints: 0
- Variables: 3844
- Time (secs):
  - LOQO: 5.1
  - LANCELOT: 4.0
  - SNOPT: *
Finite Impulse Response (FIR) Filter Design

- Audio is stored digitally in a computer as a stream of short integers: $u_k, k \in \mathbb{Z}$.

- When the music is played, these integers are used to drive the displacement of the speaker from its resting position.

- For CD quality sound, 44100 short integers get played per second per channel.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>8</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-32768</td>
<td>-23681</td>
<td>12111</td>
</tr>
<tr>
<td>2</td>
<td>-32768</td>
<td>-18449</td>
<td>17311</td>
</tr>
<tr>
<td>3</td>
<td>-30753</td>
<td>-11025</td>
<td>21311</td>
</tr>
<tr>
<td>4</td>
<td>-28865</td>
<td>-6913</td>
<td>23055</td>
</tr>
<tr>
<td>5</td>
<td>-29105</td>
<td>-4337</td>
<td>23519</td>
</tr>
<tr>
<td>6</td>
<td>-29201</td>
<td>-1329</td>
<td>25247</td>
</tr>
<tr>
<td>7</td>
<td>-26513</td>
<td>1743</td>
<td>27535</td>
</tr>
<tr>
<td>8</td>
<td>6223</td>
<td>29471</td>
<td></td>
</tr>
</tbody>
</table>
FIR Filter Design—Continued

- A finite impulse response (FIR) filter takes as input a digital signal and convolves this signal with a finite set of fixed numbers $h_{-n}, \ldots, h_n$ to produce a filtered output signal:

$$y_k = \sum_{i=-n}^{n} h_i u_{k-i}.$$ 

- Sparing the details, the output power at frequency $\nu$ is given by

$$|H(\nu)|$$

where

$$H(\nu) = \sum_{k=-n}^{n} h_k e^{2\pi ik\nu},$$

- Similarly, the mean squared deviation from a flat frequency response over a frequency range, say $\mathcal{L} \subset [0, 1]$, is given by

$$\frac{1}{|\mathcal{L}|} \int_{\mathcal{L}} |H(\nu) - 1|^2 d\nu.$$
Filter Design: Woofer, Midrange, Tweeter

minimize $\rho$

subject to $\int_0^1 (H_w(\nu) + H_m(\nu) + H_t(\nu) - 1)^2 d\nu \leq \epsilon$

$\left(\frac{1}{|W|} \int_W H^2_w(\nu) d\nu\right)^{1/2} \leq \rho \quad W = [.2, .8]$

$\left(\frac{1}{|M|} \int_M H^2_m(\nu) d\nu\right)^{1/2} \leq \rho \quad M = [.4, .6] \cup [.9, .1]$

$\left(\frac{1}{|T|} \int_T H^2_t(\nu) d\nu\right)^{1/2} \leq \rho \quad T = [.7, .3]$

where

$H_i(\nu) = h_i(0) + 2 \sum_{k=1}^{n-1} h_i(k) \cos(2\pi k\nu), \quad i = W, M, T$

$h_i(k) = \text{filter coefficients, i.e., decision variables}$
Specific Example

filter length: \( n = 14 \)

integral discretization: \( N = 1000 \)

constraints \( = 4 \)

variables \( = 43 \)

time (secs)

<table>
<thead>
<tr>
<th>Solver</th>
<th>Time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOQO</td>
<td>79</td>
</tr>
<tr>
<td>MINOS</td>
<td>164</td>
</tr>
<tr>
<td>LANCELOT</td>
<td>3401</td>
</tr>
<tr>
<td>SNOPT</td>
<td>35</td>
</tr>
</tbody>
</table>

Ref: J.O. Coleman, U.S. Naval Research Laboratory,
CISS98 paper available: [engr.umbc.edu/~jeffc/pubs/abstracts/ciss98.html](http://engr.umbc.edu/~jeffc/pubs/abstracts/ciss98.html)

Click here for demo
Specific Example

filter length: $n = 14$
integral discretization: $N = 1000$

constraints 4
variables 43
time (secs)
    LOQO 79
    MINOS 164
    LANCELOT 3401
    SNOPT 35

Ref: J.O. Coleman, U.S. Naval Research Laboratory,
CISS98 paper available: enr.umbc.edu/~jeffc/pubs/abstracts/ciss98.html

Click here for demo
Examples: Nonconvex Optimization Models
Celestial Mechanics—Periodic Orbits

- Find periodic orbits for the planar gravitational $n$-body problem.
- Minimize action:
  \[
  \int_0^{2\pi} (K(t) - P(t)) dt,
  \]
  where $K(t)$ is kinetic energy,
  \[
  K(t) = \frac{1}{2} \sum_i \left( \dot{x}_i^2(t) + \dot{y}_i^2(t) \right),
  \]
  and $P(t)$ is potential energy,
  \[
  P(t) = - \sum_{i<j} \frac{1}{\sqrt{(x_i(t)-x_j(t))^2 + (y_i(t)-y_j(t))^2}}.
  \]
- Subject to periodicity constraints:
  \[
  x_i(2\pi) = x_i(0), \quad y_i(2\pi) = y_i(0).
  \]
Celestial Mechanics—Periodic Orbits

- Find periodic orbits for the planar gravitational $n$-body problem.
- Minimize action:
  \[
  \int_{0}^{2\pi} (K(t) - P(t)) dt,
  \]
- where $K(t)$ is kinetic energy,
  \[
  K(t) = \frac{1}{2} \sum_{i} \left( \dot{x}_{i}^{2}(t) + \dot{y}_{i}^{2}(t) \right),
  \]
- and $P(t)$ is potential energy,
  \[
  P(t) = -\sum_{i<j} \frac{1}{\sqrt{(x_{i}(t) - x_{j}(t))^{2} + (y_{i}(t) - y_{j}(t))^{2}}}.\]
- Subject to periodicity constraints:
  \[
  x_{i}(2\pi) = x_{i}(0), \quad y_{i}(2\pi) = y_{i}(0).
  \]
Celestial Mechanics—Periodic Orbits

- Find periodic orbits for the planar gravitational $n$-body problem.

- Minimize action:

$$\int_{0}^{2\pi} (K(t) - P(t)) dt,$$

- where $K(t)$ is kinetic energy,

$$K(t) = \frac{1}{2} \sum_{i} \left( \dot{x}_{i}^{2}(t) + \dot{y}_{i}^{2}(t) \right),$$

- and $P(t)$ is potential energy,

$$P(t) = -\sum_{i<j} \frac{1}{\sqrt{(x_{i}(t) - x_{j}(t))^{2} + (y_{i}(t) - y_{j}(t))^{2}}}.$$

- Subject to periodicity constraints:

$$x_{i}(2\pi) = x_{i}(0), \quad y_{i}(2\pi) = y_{i}(0).$$
Celestial Mechanics—Periodic Orbits

- Find periodic orbits for the planar gravitational $n$-body problem.
- Minimize action:
  \[ \int_0^{2\pi} (K(t) - P(t))dt, \]
- where $K(t)$ is kinetic energy,
  \[ K(t) = \frac{1}{2} \sum_i \left( \dot{x}_i^2(t) + \dot{y}_i^2(t) \right), \]
- and $P(t)$ is potential energy,
  \[ P(t) = -\sum_{i<j} \frac{1}{\sqrt{(x_i(t) - x_j(t))^2 + (y_i(t) - y_j(t))^2}}. \]
- Subject to periodicity constraints:
  \[ x_i(2\pi) = x_i(0), \quad y_i(2\pi) = y_i(0). \]
21 Celestial Mechanics—Periodic Orbits

• Find periodic orbits for the planar gravitational \( n \)-body problem.

• Minimize action:

\[
\int_0^{2\pi} (K(t) - P(t)) dt,
\]

where \( K(t) \) is kinetic energy,

\[
K(t) = \frac{1}{2} \sum_i \left( \dot{x}_i^2(t) + \dot{y}_i^2(t) \right),
\]

and \( P(t) \) is potential energy,

\[
P(t) = -\sum_{i<j} \frac{1}{\sqrt{(x_i(t) - x_j(t))^2 + (y_i(t) - y_j(t))^2}}.
\]

• Subject to periodicity constraints:

\[
x_i(2\pi) = x_i(0), \quad y_i(2\pi) = y_i(0).
\]
Celestial Mechanics—Periodic Orbits

- Find periodic orbits for the planar gravitational $n$-body problem.

- Minimize action:

$$\int_0^{2\pi} (K(t) - P(t))dt,$$

where $K(t)$ is kinetic energy,

$$K(t) = \frac{1}{2} \sum_i \left( \dot{x}_i^2(t) + \dot{y}_i^2(t) \right),$$

and $P(t)$ is potential energy,

$$P(t) = -\sum_{i<j} \frac{1}{\sqrt{(x_i(t) - x_j(t))^2 + (y_i(t) - y_j(t))^2}}.$$

- Subject to periodicity constraints:

$$x_i(2\pi) = x_i(0), \quad y_i(2\pi) = y_i(0).$$
22 Specific Example

Orbits.mod with $n = 3$ and $(0, 2\pi)$ discretized into a 160 pieces gives the following results:

- constraints: 0
- variables: 960
- time (secs):
  - LOQO: 1.1
  - LANCELOT: 8.7
  - SNOPT: 287 (no change for last 80% of iterations)
23 Putting on an Uneven Green

Given:

- $z(x, y)$ elevation of the green.
- Starting position of the ball $(x_0, y_0)$.
- Position of hole $(x_f, y_f)$.
- Coefficient of friction $\mu$.

Find: initial velocity vector so that ball will roll to the hole and arrive with minimal speed.

Variables:

- $u(t) = (x(t), y(t), z(t))$—position as a function of time $t$.
- $v(t) = (v_x(t), v_y(t), v_z(t))$—velocity.
- $a(t) = (a_x(t), a_y(t), a_z(t))$—acceleration.
- $T$—time at which ball arrives at hole.
23 Putting on an Uneven Green

Given:

- \( z(x, y) \) elevation of the green.
- Starting position of the ball \((x_0, y_0)\).
- Position of hole \((x_f, y_f)\).
- Coefficient of friction \( \mu \).

Find: initial velocity vector so that ball will roll to the hole and arrive with minimal speed.

Variables:

- \( u(t) = (x(t), y(t), z(t)) \) —position as a function of time \( t \).
- \( v(t) = (v_x(t), v_y(t), v_z(t)) \) —velocity.
- \( a(t) = (a_x(t), a_y(t), a_z(t)) \) —acceleration.
- \( T \) —time at which ball arrives at hole.
23 Putting on an Uneven Green

Given:

- \( z(x, y) \) elevation of the green.
- Starting position of the ball \((x_0, y_0)\).
- Position of hole \((x_f, y_f)\).
- Coefficient of friction \( \mu \).

Find: initial velocity vector so that ball will roll to the hole and arrive with minimal speed.

Variables:

- \( u(t) = (x(t), y(t), z(t)) \) —position as a function of time \( t \).
- \( v(t) = (v_x(t), v_y(t), v_z(t)) \) —velocity.
- \( a(t) = (a_x(t), a_y(t), a_z(t)) \) —acceleration.
- \( T \) —time at which ball arrives at hole.
23 Putting on an Uneven Green

Given:

- \(z(x, y)\) elevation of the green.
- Starting position of the ball \((x_0, y_0)\).
- Position of hole \((x_f, y_f)\).
- Coefficient of friction \(\mu\).

Find: initial velocity vector so that ball will roll to the hole and arrive with minimal speed.

Variables:

- \(u(t) = (x(t), y(t), z(t))\) — position as a function of time \(t\).
- \(v(t) = (v_x(t), v_y(t), v_z(t))\) — velocity.
- \(a(t) = (a_x(t), a_y(t), a_z(t))\) — acceleration.
- \(T\) — time at which ball arrives at hole.
Putting—Two Approaches

- Problem can be formulated with two decision variables:

\[ v_x(0) \quad \text{and} \quad v_y(0) \]

and two constraints:

\[ x(T) = x_f \quad \text{and} \quad y(T) = y_f. \]

In this case, \( x(T), y(T) \), and the objective function are complicated functions of the two variables that can only be computed by integrating the appropriate differential equation.

- A discretization of the complete trajectory (including position, velocity, and acceleration) can be taken as variables and the physical laws encoded in the differential equation can be written as constraints.

To implement the first approach, one would need an ode integrator that provides, in addition to the quantities being sought, first and possibly second derivatives of those quantities with respect to the decision variables.

The modern trend is to follow the second approach.
Problem can be formulated with two decision variables:

\[ v_x(0) \quad \text{and} \quad v_y(0) \]

and two constraints:

\[ x(T) = x_f \quad \text{and} \quad y(T) = y_f. \]

In this case, \( x(T), y(T), \) and the objective function are complicated functions of the two variables that can only be computed by integrating the appropriate differential equation.

A discretization of the complete trajectory (including position, velocity, and acceleration) can be taken as variables and the physical laws encoded in the differential equation can be written as constraints.

To implement the first approach, one would need an ode integrator that provides, in addition to the quantities being sought, first and possibly second derivatives of those quantities with respect to the decision variables.

The modern trend is to follow the second approach.
Problem can be formulated with two decision variables:

\[ v_x(0) \quad \text{and} \quad v_y(0) \]

and two constraints:

\[ x(T) = x_f \quad \text{and} \quad y(T) = y_f. \]

In this case, \( x(T), y(T) \), and the objective function are complicated functions of the two variables that can only be computed by integrating the appropriate differential equation.

A discretization of the complete trajectory (including position, velocity, and acceleration) can be taken as variables and the physical laws encoded in the differential equation can be written as constraints.

To implement the first approach, one would need an ode integrator that provides, in addition to the quantities being sought, first and possibly second derivatives of those quantities with respect to the decision variables.

The modern trend is to follow the second approach.
Problem can be formulated with two decision variables:

\[ v_x(0) \text{ and } v_y(0) \]

and two constraints:

\[ x(T) = x_f \text{ and } y(T) = y_f. \]

In this case, \( x(T), y(T) \), and the objective function are complicated functions of the two variables that can only be computed by integrating the appropriate differential equation.

A discretization of the complete trajectory (including position, velocity, and acceleration) can be taken as variables and the physical laws encoded in the differential equation can be written as constraints.

To implement the first approach, one would need an ode integrator that provides, in addition to the quantities being sought, first and possibly second derivatives of those quantities with respect to the decision variables.

The modern trend is to follow the second approach.
Putting—Two Approaches

- Problem can be formulated with two decision variables:

\[ v_x(0) \text{ and } v_y(0) \]

and two constraints:

\[ x(T) = x_f \text{ and } y(T) = y_f. \]

In this case, \( x(T), y(T) \), and the objective function are complicated functions of the two variables that can only be computed by integrating the appropriate differential equation.

- A discretization of the complete trajectory (including position, velocity, and acceleration) can be taken as variables and the physical laws encoded in the differential equation can be written as constraints.

To implement the first approach, one would need an ode integrator that provides, in addition to the quantities being sought, first and possibly second derivatives of those quantities with respect to the decision variables.

The modern trend is to follow the second approach.
25 Putting—Continued

Objective:

\[ \text{minimize } v_x(T)^2 + v_y(T)^2. \]

Constraints:

\[ v = \dot{u} \]
\[ a = \dot{v} \]
\[ ma = N + F - mge_z \]
\[ u(0) = u_0 \quad u(T) = u_f, \]

where

- \( m \) is the mass of the golf ball.
- \( g \) is the acceleration due to gravity.
- \( e_z \) is a unit vector in the positive \( z \) direction.

and ...
Objective:

\[ \text{minimize } v_x(T)^2 + v_y(T)^2. \]

Constraints:

\[ v = \dot{u} \]
\[ a = \dot{v} \]
\[ ma = N + F - mge_z \]
\[ u(0) = u_0 \quad u(T) = u_f, \]

where

- \( m \) is the mass of the golf ball.
- \( g \) is the acceleration due to gravity.
- \( e_z \) is a unit vector in the positive \( z \) direction.

and ...
25 Putting—Continued

Objective:

\[
\text{minimize } v_x(T)^2 + v_y(T)^2.
\]

Constraints:

\[
\begin{align*}
\dot{v} &= \ddot{u} \\
\dot{a} &= \ddot{v} \\
ma &= N + F - mge_z \\
u(0) &= u_0 \quad u(T) = u_f,
\end{align*}
\]

where

- \( m \) is the mass of the golf ball.
- \( g \) is the acceleration due to gravity.
- \( e_z \) is a unit vector in the positive \( z \) direction.

and ...
25 Putting—Continued

Objective:

\[ \text{minimize } v_x(T)^2 + v_y(T)^2. \]

Constraints:

\[ v = \dot{u} \]
\[ a = \dot{v} \]
\[ ma = N + F - mge_z \]
\[ u(0) = u_0 \quad u(T) = u_f, \]

where

- \( m \) is the mass of the golf ball.
- \( g \) is the acceleration due to gravity.
- \( e_z \) is a unit vector in the positive \( z \) direction.

and ...
• $N = (N_x, N_y, N_z)$ is the normal force:

\[
N_z = m \frac{g - a_x(t) \frac{\partial z}{\partial x} - a_y(t) \frac{\partial z}{\partial y} + a_z(t)}{\left( \frac{\partial z}{\partial x} \right)^2 + \left( \frac{\partial z}{\partial y} \right)^2 + 1}
\]

\[
N_x = -\frac{\partial z}{\partial x} N_z
\]

\[
N_y = -\frac{\partial z}{\partial y} N_z
\]

• $F$ is the force due to friction:

\[
F = -\mu \|N\| \frac{v}{\|v\|}.
\]
Putting—Continued

- \( \mathbf{N} = (N_x, N_y, N_z) \) is the normal force:

\[
N_z = m \frac{g - a_x(t) \frac{\partial z}{\partial x} - a_y(t) \frac{\partial z}{\partial y} + a_z(t)}{(\frac{\partial z}{\partial x})^2 + (\frac{\partial z}{\partial y})^2 + 1} \\
N_x = -\frac{\partial z}{\partial x} N_z \\
N_y = -\frac{\partial z}{\partial y} N_z.
\]

- \( \mathbf{F} \) is the force due to friction:

\[
F = -\mu \| \mathbf{N} \| \frac{v}{\| v \|}.
\]
• \( N = (N_x, N_y, N_z) \) is the normal force:

\[
N_z = m \frac{g - a_x(t) \frac{\partial z}{\partial x} - a_y(t) \frac{\partial z}{\partial y} + a_z(t)}{(\frac{\partial z}{\partial x})^2 + (\frac{\partial z}{\partial y})^2 + 1}
\]

\[
N_x = -\frac{\partial z}{\partial x} N_z
\]

\[
N_y = -\frac{\partial z}{\partial y} N_z.
\]

• \( F \) is the force due to friction:

\[
F = -\mu ||N|| \frac{v}{||v||}.
\]
27 Putting—Specific Example

- Discretize continuous time into \( n = 200 \) discrete time points.
- Use finite differences to approximate the derivatives.

<table>
<thead>
<tr>
<th>Constraints</th>
<th>Variables</th>
<th>Time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOQO</td>
<td>597</td>
<td>14.1</td>
</tr>
<tr>
<td>LANCELOT</td>
<td>399</td>
<td>&gt; 600.0</td>
</tr>
<tr>
<td>SNOPT</td>
<td>4.1</td>
<td></td>
</tr>
</tbody>
</table>
Objective:

\[
\text{maximize } h(T);
\]

Constraints:

\[
\begin{align*}
v &= \dot{h}\\
a &= \dot{v}\\
\theta &= -cm\\
ma &= (\theta - \sigma v^2 e^{-h/h_0}) - gm\\
0 &\leq \theta \leq \theta_{\text{max}}\\
m &\geq m_{\text{min}}\\
h(0) &= 0 \quad v(0) = 0 \quad m(0) = 3
\end{align*}
\]

where

- \( \theta = \text{Thrust, } m = \text{mass} \)
- \( \theta_{\text{max}}, g, \sigma, c, \text{ and } h_0 \) are given constants
- \( h, v, a, T_h, \text{ and } m \) are functions of time \( 0 \leq t \leq T \).
Objective:

maximize $h(T)$;

Constraints:

$v = \dot{h}$

$a = \dot{v}$

$\theta = -cm$

$ma = (\theta - \sigma v^2 e^{-h/h_0}) - gm$

$0 \leq \theta \leq \theta_{\text{max}}$

$m \geq m_{\text{min}}$

$h(0) = 0 \quad v(0) = 0 \quad m(0) = 3$

where

- $\theta =$ Thrust, $m =$ mass
- $\theta_{\text{max}}, g, \sigma, c,$ and $h_0$ are given constants
- $h, v, a, T_h,$ and $m$ are functions of time $0 \leq t \leq T.$
Goddard Rocket Problem—Solution

constraints 399
variables 599

time (secs)

LOQO 5.2
LANCELOT (IL)
SNOPT (IL)