Lecture 3
Interior Point Methods
and Nonlinear Optimization

Robert J. Vanderbei

April 16, 2012

Machine Learning Summer School
La Palma

http://www.princeton.edu/~rvdb
Example: Basis Pursuit Denoising
A trade-off between two objectives:
1. Least squares regression: \( \min \frac{1}{2} \|Ax - b\|_2^2 \).
2. Sparsity of the solution as encouraged by minimizing \( \sum_j |x_j| \).

Trade-off:
\[
\min \frac{1}{2} \|Ax - b\|_2^2 + \lambda \|x\|_1.
\]

Ideal value for \( \lambda \) is unknown.

May wish to try many different values hoping to find a good one.

**Suggestion:**
- Change least-squares regression to least-absolute-value regression,
- formulate the problem as a parametric linear programming problem, and
- solve it for all values of \( \lambda \) using the parametric simplex method.

This is an important problem in machine learning.
Interior-Point Methods
What Makes LP Hard?

**Primal**

maximize \( c^T x \)

subject to \( Ax + w = b \)

\( x, w \geq 0 \)

**Dual**

minimize \( b^T y \)

subject to \( A^T y - z = c \)

\( y, z \geq 0 \)

**Complementarity Conditions**

\( x_j z_j = 0 \quad j = 1, 2, \ldots, n \)

\( w_i y_i = 0 \quad i = 1, 2, \ldots, m \)
Can't write $xz = 0$. The product $xz$ is undefined.

Instead, introduce a new notation:

$$
x = \begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_n
\end{bmatrix} \quad \implies \quad X = \begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_n
\end{bmatrix}
$$

Then the complementarity conditions can be written as:

$$
X Z e = 0 \\
W Y e = 0
$$
Optimality Conditions

\[ Ax + w = b \]
\[ A^T y - z = c \]
\[ ZX e = 0 \]
\[ WY e = 0 \]
\[ w, x, y, z \geq 0 \]

Ignore (temporarily) the nonnegativities.

2n + 2m equations in 2n + 2m unknowns.

Solve’em.

Hold on. Not all equations are linear.

*It is the nonlinearity of the complementarity conditions that makes LP fundamentally harder than solving systems of equations.*
Since we’re ignoring nonnegativities, it’s best to replace complementarity with μ-complementarity:

\[
\begin{align*}
Ax + w &= b \\
A^T y - z &= c \\
ZXe &= \mu e \\
WYe &= \mu e
\end{align*}
\]

Start with an arbitrary (positive) initial guess: \(x, y, w, z\).

Introduce step directions: \(\Delta x, \Delta y, \Delta w, \Delta z\).

Write the above equations for \(x + \Delta x, y + \Delta y, w + \Delta w, \text{ and } z + \Delta z\):

\[
\begin{align*}
A(x + \Delta x) + (w + \Delta w) &= b \\
A^T(y + \Delta y) - (z + \Delta z) &= c \\
(Z + \Delta Z)(X + \Delta X)e &= \mu e \\
(W + \Delta W)(Y + \Delta Y)e &= \mu e
\end{align*}
\]
Rearrange with “delta” variables on left and drop nonlinear terms on left:

\[
\begin{align*}
A\Delta x + \Delta w &= b - Ax - w \\
A^T \Delta y - \Delta z &= c - A^T y + z \\
Z \Delta x + X \Delta z &= \mu e - Z X e \\
W \Delta y + Y \Delta w &= \mu e - W Y e 
\end{align*}
\]

This is a \textit{linear} system of \(2m + 2n\) equations in \(2m + 2n\) unknowns.

Solve’em.

Dampen the step lengths, if necessary, to maintain positivity.

Step to a new point:

\[
\begin{align*}
x &\leftarrow x + \theta \Delta x \\
y &\leftarrow y + \theta \Delta y \\
w &\leftarrow w + \theta \Delta w \\
z &\leftarrow z + \theta \Delta z
\end{align*}
\]

(\(\theta\) is the scalar damping factor).
Recall equations

\[
\begin{align*}
A\Delta x + \Delta w &= b - Ax - w \\
A^T\Delta y - \Delta z &= c - A^Ty + z \\
Z\Delta x + X\Delta z &= \mu e - ZXe \\
W\Delta y + Y\Delta w &= \mu e - WYe
\end{align*}
\]

Solve for $\Delta z$

\[
\Delta z = X^{-1}(\mu e - ZXe - Z\Delta x)
\]

and for $\Delta w$

\[
\Delta w = Y^{-1}(\mu e - WYe - W\Delta y).
\]

Eliminate $\Delta z$ and $\Delta w$ from first two equations:

\[
\begin{align*}
A\Delta x - Y^{-1}W\Delta y &= b - Ax - \mu Y^{-1}e \\
A^T\Delta y + X^{-1}Z\Delta x &= c - A^Ty + \mu X^{-1}e
\end{align*}
\]
Pick a smaller value of $\mu$ for the next iteration.

Repeat from beginning until current solution satisfies, within a tolerance, optimality conditions:

**primal feasibility** $b - Ax - w = 0$.

**dual feasibility** $c - A^T y + z = 0$.

**duality gap** $b^T y - c^T x = 0$.

**Theorem.**

- Primal infeasibility gets smaller by a factor of $1 - \theta$ at every iteration.
- Dual infeasibility gets smaller by a factor of $1 - \theta$ at every iteration.
- If primal and dual are feasible, then duality gap decreases by a factor of $1 - \theta$ at every iteration (if $\mu = 0$, slightly slower convergence if $\mu > 0$).
Hard/impossible to “do” an interior-point method by hand.

Yet, easy to program on a computer (solving large systems of equations is routine).

LOQO implements an interior-point method.

Setting option loqo_options ’verbose=2’ in AMPL produces the following “typical” output:
LOQO Output

variables: non-neg 1350, free 0, bdd 0, total 1350
constraints: eq 146, ineq 0, ranged 0, total 146
nonzeros: A 5288, Q 0
nonzeros: L 7953, arith_ops 101444

<table>
<thead>
<tr>
<th>Iter</th>
<th>Primal</th>
<th>Dual</th>
<th>Sig</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Obj Value</td>
<td>Infeas</td>
<td>Obj Value</td>
</tr>
<tr>
<td>1</td>
<td>-7.8000000e+03</td>
<td>1.55e+03</td>
<td>5.5076028e-01</td>
</tr>
<tr>
<td>2</td>
<td>2.6725737e+05</td>
<td>7.84e+01</td>
<td>1.0917132e+00</td>
</tr>
<tr>
<td>3</td>
<td>1.1880365e+05</td>
<td>3.92e+00</td>
<td>4.5697310e-01</td>
</tr>
<tr>
<td>4</td>
<td>6.7391043e+03</td>
<td>2.22e-01</td>
<td>7.2846138e-01</td>
</tr>
<tr>
<td>5</td>
<td>9.5202841e+02</td>
<td>3.12e-02</td>
<td>5.4810461e+00</td>
</tr>
<tr>
<td>6</td>
<td>2.1095320e+02</td>
<td>6.03e-03</td>
<td>2.7582307e+01</td>
</tr>
<tr>
<td>7</td>
<td>8.5669013e+01</td>
<td>1.36e-03</td>
<td>4.2343105e+01</td>
</tr>
<tr>
<td>8</td>
<td>5.8494756e+01</td>
<td>3.42e-04</td>
<td>4.6750024e+01</td>
</tr>
<tr>
<td>9</td>
<td>5.1228667e+01</td>
<td>8.85e-05</td>
<td>4.7875326e+01</td>
</tr>
<tr>
<td>10</td>
<td>4.9466277e+01</td>
<td>2.55e-05</td>
<td>4.8617380e+01</td>
</tr>
<tr>
<td>11</td>
<td>4.8792989e+01</td>
<td>1.45e-06</td>
<td>4.8736603e+01</td>
</tr>
<tr>
<td>12</td>
<td>4.8752154e+01</td>
<td>7.26e-08</td>
<td>4.8749328e+01</td>
</tr>
<tr>
<td>13</td>
<td>4.8750108e+01</td>
<td>3.63e-09</td>
<td>4.8749966e+01</td>
</tr>
<tr>
<td>14</td>
<td>4.8750005e+01</td>
<td>1.81e-10</td>
<td>4.8749998e+01</td>
</tr>
<tr>
<td>15</td>
<td>4.8750000e+01</td>
<td>9.07e-12</td>
<td>4.8750000e+01</td>
</tr>
</tbody>
</table>

OPTIMAL SOLUTION FOUND
A Generalizable Framework

Start with an optimization problem—in this case LP:

\[
\begin{align*}
\text{maximize} & \quad c^T x \\
\text{subject to} & \quad Ax \leq b \\
& \quad x \geq 0
\end{align*}
\]

Use slack variables to make all inequality constraints into nonnegativities:

\[
\begin{align*}
\text{maximize} & \quad c^T x \\
\text{subject to} & \quad Ax + w = b \\
& \quad x, w \geq 0
\end{align*}
\]

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

\[
\begin{align*}
\text{maximize} & \quad c^T x + \mu \sum_j \log x_j + \mu \sum_i \log w_i \\
\text{subject to} & \quad Ax + w = b
\end{align*}
\]
Incorporate the equality constraints into the objective using *Lagrange multipliers*:

\[
L(x, w, y) = c^T x + \mu \sum_j \log x_j + \mu \sum_i \log w_i + y^T (b - Ax - w)
\]

Set derivatives to zero:

\[
c + \mu X^{-1} e - A^T y = 0 \quad \text{(deriv wrt } x) \\
\mu W^{-1} e - y = 0 \quad \text{(deriv wrt } w) \\
b - Ax - w = 0 \quad \text{(deriv wrt } y) 
\]

Introduce *dual complementary variables*:

\[
z = \mu X^{-1} e
\]

Rewrite system:

\[
c + z - A^T y = 0 \\
XZe = \mu e \\
WYe = \mu e \\
b - Ax - w = 0
\]
Logarithmic Barrier Functions

Plots of $\mu \log x$ for various values of $\mu$:
Lagrange Multipliers

maximize \( f(x) \)
subject to \( g(x) = 0 \)

maximize \( f(x) \)
subject to \( g_1(x) = 0 \)
\( g_2(x) = 0 \)
Nonlinear Optimization
Outline

• Algorithm
  – Basic Paradigm
  – Step-Length Control
  – Diagonal Perturbation
The Interior-Point Algorithm
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:

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

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

\[
\begin{align*}
\text{minimize} & \quad f(x) - \mu \sum_{i=1}^{m} \log(w_i) \\
\text{subject to} & \quad h(x) - w = 0
\end{align*}
\]
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:

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

- Rewrite system:

\[
\nabla f(x) - \nabla h(x)^T y = 0 \\
WYe = \mu e \\
h(x) - w = 0
\]
Apply Newton’s Method

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

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

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 $y_i \geq 0$. 
• 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:

\[
x^{(k+1)} = x^{(k)} + \alpha^{(k)} \Delta x^{(k)}
\]
\[
w^{(k+1)} = w^{(k)} + \alpha^{(k)} \Delta w^{(k)}
\]
\[
y^{(k+1)} = y^{(k)} + \alpha^{(k)} \Delta y^{(k)}
\]
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 \textit{convex} if

\begin{itemize}
  \item \( h_i \)'s in equality constraints are affine;
  \item \( h_i \)'s in inequality constraints are concave;
  \item \( f \) is convex;
\end{itemize}

NLP is \textit{smooth} if

\begin{itemize}
  \item All are twice continuously differentiable.
\end{itemize}
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.
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** 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.
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 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.
Theorem 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.
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*. 
AMPL Info

• The language is called AMPL, which stands for A Mathematical Programming Language.

• The “official” document describing the language is a book called “AMPL” by Fourer, Gay, and Kernighan. Amazon.com sells it for $78.01.

• There are also online tutorials:
  – http://www2.isye.gatech.edu/~jswann/teaching/AMPLTutorial.pdf
  – Google: “AMPL tutorial” for several more.
NEOS Info

NEOS is the *Network Enabled Optimization Server* supported by our federal government and located at *Argonne National Lab*.

To submit an AMPL model to NEOS...

- visit [http://www.neos-server.org/neos/](http://www.neos-server.org/neos/),
- click on the icon,
- scroll down to the *Nonlinearly Constrained Optimization* list,
- click on LOQO [AMPL input],
- scroll down to *Model File:*,
- click on *Choose File*,
- select a file from your computer that contains an AMPL model,
- scroll down to *e-mail address:*,
- type in your email address, and
- click *Submit to NEOS*.

Piece of cake!
The Homogeneous Self-Dual Method
The Homogeneous Self-Dual Problem

Primal-Dual Pair

maximize \( c^T x \) subject to \( Ax \leq b \) \( x \geq 0 \)

minimize \( b^T y \) subject to \( A^T y \geq c \) \( y \geq 0 \)

Homogeneous Self-Dual Problem

maximize \( 0 \) subject to \( -A^T y + c\phi \leq 0 \)
\( Ax \geq 0 \)
\( -c^T x + b^T y \leq 0 \)
\( x, y, \phi \geq 0 \)
In Matrix Notation

maximize \[ 0 \]
subject to
\[
\begin{bmatrix}
0 & -A^T & c \\
A & 0 & -b \\
-c^T & b^T & 0
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
\phi
\end{bmatrix}
\leq
\begin{bmatrix}
0 \\
0 \\
0
\end{bmatrix}
\]
\[ x, y, \phi \geq 0. \]

HSD is self-dual (constraint matrix is skew symmetric).

HSD is feasible \((x = 0, y = 0, \phi = 0)\).

HSD is homogeneous—i.e., multiplying a feasible solution by a positive constant yields a new feasible solution.

Any feasible solution is optimal.

If \(\phi\) is a null variable, then either primal or dual is infeasible (see text).
Theorem. Let \((x, y, \phi)\) be a solution to HSD. If \(\phi > 0\), then

- \(x^* = x/\phi\) is optimal for primal, and
- \(y^* = y/\phi\) is optimal for dual.

Proof.

\(x^*\) is primal feasible—obvious.
\(y^*\) is dual feasible—obvious.

Weak duality theorem implies that \(c^T x^* \leq b^T y^*\).

3rd HSD constraint implies reverse inequality.

Primal feasibility, plus dual feasibility, plus no gap implies optimality.
In New Notation:

maximize 0
subject to \( Ax + z = 0 \)
\( x, z \geq 0 \)
More Notation

Infeasibility: \( \rho(x, z) = Ax + z \)
Complementarity: \( \mu(x, z) = \frac{1}{n} x^T z \)

Nonlinear System

\[
A(x + \Delta x) + (z + \Delta z) = \delta(Ax + z) \\
(X + \Delta X)(Z + \Delta Z)e = \delta \mu(x, z)e
\]

Linearized System

\[
A\Delta x + \Delta z = -(1 - \delta) \rho(x, z) \\
Z \Delta x + X \Delta z = \delta \mu(x, z)e - XZe
\]
**Algorithm**

Solve linearized system for $(\Delta x, \Delta z)$.

Pick step length $\theta$.

Step to a new point:

$$\bar{x} = x + \theta \Delta x, \quad \bar{z} = z + \theta \Delta z.$$  

**Even More Notation**

$$\bar{\rho} = \rho(\bar{x}, \bar{z}), \quad \bar{\mu} = \mu(\bar{x}, \bar{z})$$
Theorem 2

1. $\Delta z^T \Delta x = 0$.

2. $\bar{\rho} = (1 - \theta + \theta \delta) \rho$.

3. $\bar{\mu} = (1 - \theta + \theta \delta) \mu$.

4. $\bar{X} \bar{Z} e - \bar{\mu} e = (1 - \theta)(X Z e - \mu e) + \theta^2 \Delta X \Delta Z e$.

Proof.

1. Tedious but not hard (see text).

2. 

   \[
   \begin{align*}
   \bar{\rho} & = A(x + \theta \Delta x) + (z + \theta \Delta z) \\
   & = Ax + z + \theta(A \Delta x + \Delta z) \\
   & = \rho - \theta(1 - \delta) \rho \\
   & = (1 - \theta + \theta \delta) \rho.
   \end{align*}
   \]
3.

\[ \bar{x}^T \bar{z} = (x + \theta \Delta x)^T(z + \theta \Delta z) \]
\[ = x^T z + \theta(z^T \Delta x + x^T \Delta z) + \theta^2 \Delta x^T \Delta z \]
\[ = x^T z + \theta e^T(\delta \mu e - XZe) \]
\[ = (1 - \theta + \theta \delta)x^T z. \]

Now, just divide by \( n \).

4.

\[ \bar{X} \bar{Z} e - \bar{\mu} e = (X + \theta \Delta X)(Z + \theta \Delta Z)e - (1 - \theta + \theta \delta)\mu e \]
\[ = XZe - \mu e + \theta(X \Delta z + Z \Delta x + (1 - \delta)\mu e) + \theta^2 \Delta X \Delta Ze \]
\[ = (1 - \theta)(XZe - \mu e) + \theta^2 \Delta X \Delta Ze. \]
Neighborhoods of \( \{ (x, z) > 0 : x_1 z_1 = x_2 z_2 = \cdots = x_n z_n \} \)

\[
\mathcal{N}(\beta) = \{ (x, z) > 0 : \| X Z e - \mu(x, z) e \| \leq \beta \mu(x, z) \}
\]

Note: \( \beta < \beta' \) implies \( \mathcal{N}(\beta) \subset \mathcal{N}(\beta') \).

Predictor-Corrector Algorithm

**Odd Iterations–Predictor Step**

Assume \( (x, z) \in \mathcal{N}(1/4) \).

Compute \( (\Delta x, \Delta z) \) using \( \delta = 0 \).

Compute \( \theta \) so that \( (\bar{x}, \bar{z}) \in \mathcal{N}(1/2) \).

**Even Iterations–Corrector Step**

Assume \( (x, z) \in \mathcal{N}(1/2) \).

Compute \( (\Delta x, \Delta z) \) using \( \delta = 1 \).

Put \( \theta = 1 \).
Predictor-Corrector Algorithm

In Complementarity Space

Let

\[ u_j = x_j z_j \quad j = 1, 2, \ldots, n. \]
Well-Definedness of Algorithm

Must check that preconditions for each iteration are met.

**Technical Lemma.**

1. If $\delta = 0$, then $\|\Delta X \Delta Z e\| \leq \frac{n}{2} \mu$.

2. If $\delta = 1$ and $(x, z) \in \mathcal{N}(\beta)$, then $\|\Delta X \Delta Z e\| \leq \frac{\beta^2}{1-\beta} \mu / 2$.

**Proof.** Tedious *and* tricky. See text.
**Theorem.**

1. After a predictor step, \((\bar{x}, \bar{z}) \in \mathcal{N}(1/2)\) and \(\bar{\mu} = (1 - \theta)\mu\).
2. After a corrector step, \((\bar{x}, \bar{z}) \in \mathcal{N}(1/4)\) and \(\bar{\mu} = \mu\).

**Proof.**

1. \((\bar{x}, \bar{z}) \in \mathcal{N}(1/2)\) by definition of \(\theta\).
   \[
   \bar{\mu} = (1 - \theta)\mu \text{ since } \delta = 0.
   \]

2. \(\theta = 1\) and \(\beta = 1/2\). Therefore,
   \[
   \|\bar{X}\bar{Z}e - \bar{\mu}e\| = \|\Delta X \Delta Z e\| \leq \mu/4.
   \]
   Need to show also that \((\bar{x}, \bar{z}) > 0\). Intuitively clear (see earlier picture) but proof is tedious. See text.
Complexity Analysis

Progress toward optimality is controlled by the stepsize $\theta$.

**Theorem.** In predictor steps, $\theta \geq \frac{1}{2\sqrt{n}}$.

**Proof.**

Consider taking a step with step length $t \leq 1/2\sqrt{n}$:

$$x(t) = x + t\Delta x, \quad z(t) = z + t\Delta z.$$ 

From earlier theorems and lemmas,

$$\|X(t)Z(t)e - \mu(t)e\| \leq (1 - t)\|XZe - \mu e\| + t^2\|\Delta X\Delta Ze\|$$

$$\leq (1 - t)\frac{\mu}{4} + t^2\frac{n\mu}{2}$$

$$\leq (1 - t)\frac{\mu}{4} + \frac{\mu}{8}$$

$$\leq (1 - t)\frac{\mu}{4} + (1 - t)\frac{\mu}{4}$$

$$= \frac{\mu(t)}{2}.$$ 

Therefore $(x(t), z(t)) \in \mathcal{N}(1/2)$ which implies that $\theta \geq 1/2\sqrt{n}$. 
Since
$$\mu^{(2k)} = (1 - \theta^{(2k-1)})(1 - \theta^{(2k-3)}) \cdots (1 - \theta^{(1)})\mu^{(0)}$$
and $\mu^{(0)} = 1$, we see from the previous theorem that
$$\mu^{(2k)} \leq \left(1 - \frac{1}{2\sqrt{n}}\right)^k.$$

Hence, to get a small number, say $2^{-L}$, as an upper bound for $\mu^{(2k)}$ it suffices to pick $k$ so that:
$$\left(1 - \frac{1}{2\sqrt{n}}\right)^k \leq 2^{-L}.$$

This inequality is implied by the following simpler one:
$$k \geq 2 \log(2)L\sqrt{n}.$$

Since the number of iterations is $2k$, we see that $4\log(2)L\sqrt{n}$ iterations will suffice to make the final value of $\mu$ be less than $2^{-L}$.

Of course,
$$\rho^{(k)} = \mu^{(k)}\rho^{(0)}$$
so the same bounds guarantee that the final infeasibility is small too.
Back to Original Primal-Dual Setting

Just a final remark: If primal and dual problems are feasible, then algorithm will produce a solution to HSD with $\phi > 0$ from which a solution to original problem can be extracted. See text for details.