An Interior-Point Algorithm with Inexact Step Computations for Large-scale Optimization

Frank E. Curtis, Lehigh University
Olaf Schenk, University of Basel
Andreas Wächter, IBM T. J. Watson Research

11th Copper Mountain Conference on Iterative Methods

April 5, 2010
Outline

Introduction

Interior-Point with Inexact Steps

Numerical Results

Summary and Future Work
Outline

Introduction

Interior-Point with Inexact Steps

Numerical Results

Summary and Future Work
Large-scale constrained optimization

Consider large-scale problems of the form

$$\begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad c^E(x) = 0 \\
& \quad c^I(x) \geq 0
\end{align*}$$
Large-scale constrained optimization

Consider large-scale problems of the form

\[
\begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad c^E(x) = 0 \\
& \quad c^I(x) \geq 0
\end{align*}
\]

- True problem of interest is infinite-dimensional
- Equality constraints include a discretized PDE
- \(x = (y, u)\) is composed of states \(y\) and controls \(u\)
- Inequality constraints include control (and state?) bounds
Strengths

We propose an algorithm for large-scale nonlinear optimization:

- It can handle ill-conditioned/rank-deficient problems
- It can handle nonconvex problems
- Inexactness is allowed and controlled with loose conditions
- The conditions are implementable (in fact, implemented)
- The algorithm is globally convergent
- It can handle problems with control and state constraints
- Numerical results are very encouraging so far
Weaknesses

Aim to have an algorithm for PDE-constrained optimization, but so far:

- We solve for a single discretization
- We use finite-dimensional norms
- Our implementation does not exploit structure
- We need further experimentation on interesting problems
Weaknesses

Aim to have an algorithm for PDE-constrained optimization, but so far:

- We solve for a single discretization
- We use finite-dimensional norms
- Our implementation does not exploit structure
- We need further experimentation on interesting problems

I’ll close the talk with questions; you might have the answers!
Interior-point methods

- Add slacks to form the logarithmic-barrier subproblem

\[
\begin{align*}
\min & \quad f(x) - \mu \sum_{i \in I} \ln s^i \\
\text{s.t.} & \quad c^E(x) = 0 \\
& \quad c^I(x) = s
\end{align*}
\]

- The first-order optimality conditions are

\[
\nabla f(x) + \nabla c^E(x) \lambda^E + \nabla c^I(x) \lambda^I = 0 \\
-\mu S^{-1} e - \lambda^I = 0 \\
c^E(x) = 0 \\
c^I(x) - s = 0
\]

along with \( s > 0 \)
Newton’s method

- Newton iteration involves the linear system

\[
\begin{bmatrix}
H_k & 0 & \nabla c_k^e & \nabla c_k^T \\
0 & \mu S_k^{-2} & 0 & -I \\
\nabla c_k^T & 0 & 0 & 0 \\
\nabla c_k^T & -I & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
d^x_k \\
d^s_k \\
\delta_k^e \\
\delta_k^T \\
\end{bmatrix}
= -
\begin{bmatrix}
\nabla f_k + \nabla c_k^e \lambda_k^e + \nabla c_k^T \lambda_k^T \\
-\mu S_k^{-1} e - \lambda_k^T \\
\nabla c_k^e \\
\nabla c_k^T - s_k \\
\end{bmatrix}
\]

- Search direction computation followed by a line search
Usual questions

- How do we ensure global convergence?
- How do we solve ill-conditioned problems?
- How do we handle nonconvexity?
Usual answers

- How do we ensure global convergence?
  - KKT conditions (convex case)
  - Merit/penalty function
  - Filter

- How do we solve ill-conditioned problems?
  - Matrix modifications
  - Trust regions

- How do we handle nonconvexity?
  - Matrix modifications
  - Trust regions
More questions

For large-scale problems:
  ▶ What if the derivative matrices cannot be stored?
  ▶ What if the derivative matrices cannot be factored?

We can use iterative in place of direct methods:
  ▶ Can we allow inexactness?
  ▶ How do we ensure global convergence, handle ill-conditioning, and handle nonconvexity if solutions are inexact?
Outline

Introduction

Interior-Point with Inexact Steps

Numerical Results

Summary and Future Work
Scaling and slack reset

- We begin by scaling the Newton system

\[
\begin{bmatrix}
H_k & 0 & \nabla c_k^\mathcal{E} & \nabla c_k^\mathcal{I} \\
0 & \Omega_k & 0 & -S_k \\
\nabla c_k^\mathcal{E}^T & 0 & 0 & 0 \\
\nabla c_k^\mathcal{I}^T & -S_k & 0 & 0
\end{bmatrix}
\begin{bmatrix}
d^x_k \\
\tilde{d}_k^s \\
d_k^\mathcal{E} \\
d_k^\mathcal{I}
\end{bmatrix}
= -
\begin{bmatrix}
\nabla f_k + \nabla c_k^\mathcal{E} \lambda_k^\mathcal{E} + \nabla c_k^\mathcal{I} \lambda_k^\mathcal{I} \\
-\mu e - S_k \lambda_k^\mathcal{I} \\
c_k^\mathcal{E} \\
c_k^\mathcal{I} - s_k
\end{bmatrix}
\]

- Primal-dual matrix now has nicer properties
- The use of a slack reset

\[s_k \geq \max\{0, c^\mathcal{I}(x_k)\}\]

allows easier infeasibility detection
Rank deficiency

If the constraint Jacobian is singular or ill-conditioned

- The system may be inconsistent
- The search directions \((d^x_k, \tilde{d}^s_k, \delta^E_k, \delta^I_k)\) may blow up
- The line search may break down
Matrix modifications

\[
\begin{bmatrix}
H_k & 0 & \nabla c_k^e & \nabla c_k^\tau \\
0 & \Omega_k & 0 & -S_k \\
\nabla c_k^{eT} & 0 & -\xi I & 0 \\
\nabla c_k^{\tau T} & -S_k & 0 & -\xi I
\end{bmatrix}
\begin{bmatrix}
d_k^x \\
\tilde{d}_k^s \\
d_k^\delta^e \\
\delta_k^\tau
\end{bmatrix}
= -
\begin{bmatrix}
\nabla f_k + \nabla c_k^e \lambda_k^e + \nabla c_k^\tau \lambda_k^\tau \\
-\mu e - S_k \lambda_k^\tau \\
c_k^e \\
c_k^\tau - s_k
\end{bmatrix}
\]
Matrix modifications

$$\begin{bmatrix}
H_k & 0 & \nabla c_k^E & \nabla c_k^I \\
0 & \Omega_k & 0 & -S_k \\
\nabla c_k^E^T & 0 & -\xi I & 0 \\
\nabla c_k^I^T & -S_k & 0 & -\xi I \\
\end{bmatrix}
\begin{bmatrix}
d_k^x \\
d_k^s \\
d_k^\delta^E \\
d_k^\delta^I \\
\end{bmatrix}
= -
\begin{bmatrix}
\nabla f_k + \nabla c_k^E \lambda_k^E + \nabla c_k^I \lambda_k^I \\
-\mu e - S_k \lambda_k^I \\
c_k^E \\
c_k^I - s_k \\
\end{bmatrix}$$

However, without matrix factorizations (i.e., no idea of the inertia)

- When should this modification be performed?
- What value should $\xi$ take? How large?
- How do we ensure that in the end we solve the right problem?
Failure of line search methods

- Recall the counter-example of Wächter and Biegler (2000)
Step decomposition
Normal step

\[
\min \frac{1}{2} \left\| \begin{bmatrix} c^E_k \\ c^I_k \end{bmatrix}_k - s_k \right\| + \left\| \begin{bmatrix} \nabla c^E_k^T & 0 \\ \nabla c^I_k^T & -S_k \end{bmatrix} \begin{bmatrix} v^x_k \\ v^s_k \end{bmatrix} \right\|^2 \\
\text{s.t.} \left\| \begin{bmatrix} v^x_k \\ v^s_k \end{bmatrix} \right\| \leq \omega \left\| \begin{bmatrix} \nabla c^E_k \\ 0 \\ \nabla c^I_k \end{bmatrix}_k \begin{bmatrix} c^E_k \\ -S_k \end{bmatrix}_k \right\|
\]

- QP w/ trust region constraint
- Trust region radius is zero at first-order minimizers of infeasibility
- Radius updates automatically
- Solve w/ CG or inexact dogleg
Tangential step
Tangential step
Nonconvexity

- During primal-dual step computation, **convexify** the Hessian

\[
\begin{bmatrix}
H_k + \xi I & 0 & \nabla c_k^E & \nabla c_k^T \\
0 & \Omega_k + \xi I & 0 & -S_k \\
\nabla c_k^E^T & 0 & 0 & 0 \\
\nabla c_k^T^T & -S_k & 0 & 0
\end{bmatrix}
\]

(i.e. increase \(\xi\)) whenever

\[
\begin{bmatrix}
u_k^x \\
\tilde{u}_k^s
\end{bmatrix}^T
\begin{bmatrix}
H_k + \xi I & 0 \\
n_0 & \Omega + \xi I
\end{bmatrix}
\begin{bmatrix}
u_k^x \\
\tilde{u}_k^s
\end{bmatrix} < \theta
\]

for some \(\psi, \theta > 0\)

- In our tests, modifications are few and early
- We avoid having to develop conditions for inexact projections
Primal-dual step computation

We can be brave and approach the full system (avoid normal step)

\[
\begin{bmatrix}
H_k & 0 & \nabla c^E_k & \nabla c^I_k \\
0 & \Omega_k & 0 & -S_k \\
\nabla c^E_k & 0 & 0 & 0 \\
\nabla c^I_k & -S_k & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
d^x_k \\
d^s_k \\
\delta^E_k \\
\delta^I_k \\
\end{bmatrix} = -
\begin{bmatrix}
\nabla f_k + \nabla c^E_k \lambda^E_k + \nabla c^I_k \lambda^I_k \\
-\mu e - S_k \lambda^I_k \\
c^E_k \\
c^I_k - s_k \\
\end{bmatrix}
\]

... or compute a normal step, then approach the perturbed system

\[
\begin{bmatrix}
H_k & 0 & \nabla c^E_k & \nabla c^I_k \\
0 & \Omega_k & 0 & -S_k \\
\nabla c^E_k & 0 & 0 & 0 \\
\nabla c^I_k & -S_k & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
d^x_k \\
d^s_k \\
\delta^E_k \\
\delta^I_k \\
\end{bmatrix} = -
\begin{bmatrix}
\nabla f_k + \nabla c^E_k \lambda^E_k + \nabla c^I_k \lambda^I_k \\
-\mu e - S_k \lambda^I_k \\
-\nabla c^E_k v^x_k \\
-\nabla c^I_k v^x_k + d^s_k \\
\end{bmatrix}
\]
Primal-dual step computation

We can be brave and approach the full system (avoid normal step)

\[
\begin{bmatrix}
H_k & 0 & \nabla c_k^E & \nabla c_k^I \\
0 & \Omega_k & 0 & -S_k \\
\nabla c_k^E^T & 0 & 0 & 0 \\
\nabla c_k^I^T & -S_k & 0 & 0
\end{bmatrix}
\begin{bmatrix}
d_k^x \\
d_k^s \\
\delta_k^E \\
\delta_k^I
\end{bmatrix}
= -
\begin{bmatrix}
\nabla f_k + \nabla c_k^E \lambda_k^E + \nabla c_k^I \lambda_k^I \\
-\mu e - S_k \lambda_k^I \\
c_k^E \\
c_k^I - s_k
\end{bmatrix}
\]

... or compute a normal step, then approach the perturbed system

\[
\begin{bmatrix}
H_k & 0 & \nabla c_k^E & \nabla c_k^I \\
0 & \Omega_k & 0 & -S_k \\
\nabla c_k^E^T & 0 & 0 & 0 \\
\nabla c_k^I^T & -S_k & 0 & 0
\end{bmatrix}
\begin{bmatrix}
d_k^x \\
d_k^s \\
\delta_k^E \\
\delta_k^I
\end{bmatrix}
= -
\begin{bmatrix}
\nabla f_k + \nabla c_k^E \lambda_k^E + \nabla c_k^I \lambda_k^I \\
-\mu e - S_k \lambda_k^I \\
-\nabla c_k^E^T v_k^x \\
-\nabla c_k^I^T v_k^x + d_k^s
\end{bmatrix}
\]

How do we allow inexact solutions?
Consistency between the direction and the merit function

- In unconstrained optimization and nonlinear equations, there is always consistency (even w/ inexact steps) between the step computation and the function that measures progress.

- In constrained optimization, however, our search direction is based on optimality conditions:

\[
\begin{bmatrix}
\nabla f(x) + \nabla c^E(x)\lambda^E + \nabla c^I(x)\lambda^I \\
-\mu S^{-1}e - \lambda^I \\
c^E(x) \\
c^I(x) - s
\end{bmatrix} = 0
\]

but we judge progress by a merit function:

\[
\phi(x, s; \pi) \triangleq f(x) - \mu \sum_{i \in I} \ln s^i + \pi \left\| \begin{bmatrix} c^E(x) \\ c^I(x) - s \end{bmatrix} \right\|
\]

- **Consistency is not automatic!** A direction that may reduce the KKT error may not be a direction of descent for the merit function.
Model reductions

- We ensure consistency by requiring model reductions
- Define the model of $\phi(x, s; \pi)$ at $(x_k, s_k)$:

$$m_k(d^x, \tilde{d}^s; \pi) \triangleq f_k + \nabla f_k^T d^x - \mu \sum_{i \in I} \ln s_k^i - \mu \tilde{d}^s$$

$$+ \pi \left( \left\| \begin{bmatrix} c_k^E \\ c_k^I - s_k \end{bmatrix} \right\| + \begin{bmatrix} \nabla c_k^E^T \\ \nabla c_k^I^T \end{bmatrix} \begin{bmatrix} 0 \\ -S_k \end{bmatrix} \left\| \begin{bmatrix} d^x \\ \tilde{d}^s \end{bmatrix} \right\| \right)$$

- $d_k$ is acceptable if

$$\Delta m_k(d_k^x, \tilde{d}_k^s; \pi) \triangleq m_k(0, 0; \pi_k) - m_k(d_k^x, \tilde{d}_k^s; \pi) \gg 0$$

- This ensures descent (and more)
Termination tests

\[
\begin{bmatrix}
H_k & 0 & \nabla c_k^e & \nabla c_k^I \\
0 & \Omega_k & 0 & -S_k \\
\nabla c_k^{eT} & 0 & 0 & 0 \\
\nabla c_k^{IT} & -S_k & 0 & 0
\end{bmatrix}
\begin{bmatrix}
d_k^x \\
d_k^s \\
d_k^e \\
d_k^I
\end{bmatrix}
= -
\begin{bmatrix}
\nabla f_k + \nabla c_k^e \lambda_k^e + \nabla c_k^I \lambda_k^I \\
-\mu e - S_k \lambda_k^I \\
\delta_k^e \\
\delta_k^I
\end{bmatrix}
+ 
\begin{bmatrix}
\rho_k^x \\
\rho_k^s \\
\rho_k^e \\
\rho_k^I
\end{bmatrix}
\]

Search direction is acceptable if

- **(TT1)** dual residual is sufficiently small, tangential component is bounded by normal component or by sufficient convexity, and model reduction is sufficiently large for current penalty parameter

- **(TT2)** dual residual is sufficiently small, tangential component is bounded by normal component or by sufficient convexity, and sufficient progress in linearized feasibility (model reduction obtained with increase in penalty parameter)

- **(TT3)** sufficient progress in reducing dual infeasibility
Interior-point algorithm with inexact step computations

(C., Schenk, and Wächter (2009))

for $k = 0, 1, 2, \ldots$

- Approximately solve for a normal step (optional?)
- Iteratively solve the primal-dual equations until TT1, TT2, or TT3 is satisfied, modifying the Hessian matrix when appropriate
- If only termination test 2 is satisfied, then increase $\pi$
- Backtrack from $\alpha_k \leftarrow 1$ to satisfy fraction-to-the-boundary and sufficient decrease conditions for the merit function $\phi$
- Update the iterate
- Reset the slacks
Convergence (inner iteration)

Assumption

The sequence \( \{(x_k, s_k, \lambda_k^E, \lambda_k^I)\} \) is contained in a convex set \( \Omega \) over which \( f, c^E, c^I \), and their first derivatives are bounded and Lipschitz continuous.

Theorem

If all limit points of the constraint Jacobians have full row rank, then

\[
\lim_{k \to \infty} \left\| \begin{bmatrix} \nabla f_k + \nabla c_k^E \lambda_k^E + \nabla c_k^I \lambda_k^I \\ -\mu e - S_k \lambda_k^I \\ c_k^E \\ c_k^I - s_k \end{bmatrix} \right\| = 0.
\]

Otherwise,

\[
\lim_{k \to \infty} \left\| \begin{bmatrix} \nabla c_k^E \\ 0 \\ -S_k \end{bmatrix} \begin{bmatrix} c_k^E \\ c_k^I - s_k \end{bmatrix} \right\| = 0
\]

and if \( \{\pi_k\} \) is bounded, then

\[
\lim_{k \to \infty} \left\| \begin{bmatrix} \nabla f_k + \nabla c_k^E \lambda_k^E + \nabla c_k^I \lambda_k^I \\ -\mu e - S_k \lambda_k^I \end{bmatrix} \right\| = 0
\]
Convergence (outer iteration)

Theorem

If the algorithm yields a sufficiently accurate solution to the barrier subproblem for each \( \{\mu_j\} \to 0 \) and if the linear independence constraint qualification (LICQ) holds at a limit point \( \bar{x} \) of \( \{x_j\} \), then there exist Lagrange multipliers \( \bar{\lambda} \) such that the first-order optimality conditions of the nonlinear program are satisfied.
Outline

Introduction

Interior-Point with Inexact Steps

Numerical Results

Summary and Future Work
Implementation details

- Incorporated in IPOPT software package (Wächter)
  - inexact_algorithm yes
- Linear systems solved with PARDISO (Schenk)
  - SQMR (Freund (1994))
- Preconditioning in PARDISO
  - incomplete multilevel factorization with inverse-based pivoting
  - stabilized by symmetric-weighted matchings
- Optimality tolerance: 1e-8
CUTEr and COPS collections

- 684 problems written in AMPL
- 580 solved successfully
- Robustness: ~85%
- Original IPOPT: ~94%
Parameter estimation for Helmholtz equation

Recover parameter $k$ based on data collected from propagating waves

<table>
<thead>
<tr>
<th>$N$</th>
<th>$n$</th>
<th>$p$</th>
<th>$q$</th>
<th># iter</th>
<th>CPU sec (per iter)</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>14724</td>
<td>13824</td>
<td>1800</td>
<td>37</td>
<td>807.823 (21.833)</td>
</tr>
<tr>
<td>64</td>
<td>56860</td>
<td>53016</td>
<td>7688</td>
<td>25</td>
<td>3741.42 (149.66)</td>
</tr>
<tr>
<td>128</td>
<td>227940</td>
<td>212064</td>
<td>31752</td>
<td>20</td>
<td>54581.8 (2729.1)</td>
</tr>
</tbody>
</table>
Boundary control

\[
\min \frac{1}{2} \int_{\Omega} (y(x) - y_t(x))^2 \, dx
\]

s.t. \[
\begin{align*}
-\nabla \cdot \left( e^{y(x)} \cdot \nabla y(x) \right) &= 20 \quad \text{in } \Omega \\
y(x) &= u(x) \quad \text{on } \partial \Omega \\
2.5 &\leq u(x) \leq 3.5 \quad \text{on } \partial \Omega
\end{align*}
\]

where \[
y_t(x) = 3 + 10x_1(x_1 - 1)x_2(x_2 - 1) \sin(2\pi x_3)
\]

<table>
<thead>
<tr>
<th>(N)</th>
<th>(n)</th>
<th>(p)</th>
<th>(q)</th>
<th># iter</th>
<th>CPU sec (per iter)</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>4096</td>
<td>2744</td>
<td>2704</td>
<td>13</td>
<td>2.8144 (0.2165)</td>
</tr>
<tr>
<td>32</td>
<td>32768</td>
<td>27000</td>
<td>11536</td>
<td>13</td>
<td>103.65 (7.9731)</td>
</tr>
<tr>
<td>64</td>
<td>262144</td>
<td>238328</td>
<td>47632</td>
<td>14</td>
<td>5332.3 (380.88)</td>
</tr>
</tbody>
</table>

Original IPOPT with \(N = 32\) requires 238 seconds per iteration
Hyperthermia treatment planning

\[
\min \frac{1}{2} \int_{\Omega} (y(x) - y_t(x))^2 dx
\]

\[
\begin{align*}
-\Delta y(x) - 10(y(x) - 37) &= u^* M(x) u & \text{in } \Omega \\
37.0 &\leq y(x) &\leq 37.5 &\text{on } \partial \Omega \\
42.0 &\leq y(x) &\leq 44.0 &\text{in } \Omega_0
\end{align*}
\]

where

\[ u_j = a_j e^{i\phi_j}, \quad M_{jk}(x) = \langle E_j(x), E_k(x) \rangle, \quad E_j = \sin(jx_1 x_2 x_3 \pi) \]

<table>
<thead>
<tr>
<th>(N)</th>
<th>(n)</th>
<th>(p)</th>
<th>(q)</th>
<th># iter</th>
<th>CPU sec (per iter)</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>4116</td>
<td>2744</td>
<td>2994</td>
<td>68</td>
<td>22.893 (0.3367)</td>
</tr>
<tr>
<td>32</td>
<td>32788</td>
<td>27000</td>
<td>13034</td>
<td>51</td>
<td>3055.9 (59.920)</td>
</tr>
</tbody>
</table>

Original IPOPT with \(N = 32\) requires 408 seconds per iteration
Groundwater modeling

\[
\begin{align*}
\min \ & \frac{1}{2} \int_{\Omega} (y(x) - y_t(x))^2 \, dx + \frac{1}{2} \alpha \int_{\Omega} \left[ \beta (u(x) - u_t(x))^2 + |\nabla (u(x) - u_t(x))|^2 \right] \, dx \\
\text{s.t.} \ & \\
\ & \quad \quad -\nabla \cdot (e^{u(x)} \cdot \nabla y_i(x)) = q_i(x) \quad \text{in} \ \Omega, \quad i = 1, \ldots, 6 \\
\ & \quad \quad \nabla y_i(x) \cdot n = 0 \quad \text{on} \ \partial \Omega \\
\ & \quad \quad \int_{\Omega} y_i(x) \, dx = 0, \quad i = 1, \ldots, 6 \\
\ & \quad \quad -1 \leq u(x) \leq 2 \quad \text{in} \ \Omega 
\end{align*}
\]

where

\[ q_i = 100 \sin(2\pi x_1) \sin(2\pi x_2) \sin(2\pi x_3) \]

<table>
<thead>
<tr>
<th>(N)</th>
<th>(n)</th>
<th>(p)</th>
<th>(q)</th>
<th># iter</th>
<th>CPU sec (per iter)</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>28672</td>
<td>24576</td>
<td>8192</td>
<td>18</td>
<td>206.416 (11.4676)</td>
</tr>
<tr>
<td>32</td>
<td>229376</td>
<td>196608</td>
<td>65536</td>
<td>20</td>
<td>1963.64 (98.1820)</td>
</tr>
<tr>
<td>64</td>
<td>1835008</td>
<td>1572864</td>
<td>524288</td>
<td>21</td>
<td>134418. (6400.85)</td>
</tr>
</tbody>
</table>

Original IPOPT with \(N = 32\) requires approx. 20 \textbf{hours} for the first iteration
Outline

Introduction

Interior-Point with Inexact Steps

Numerical Results

Summary and Future Work
Summary

We proposed an algorithm for large-scale nonlinear optimization:

- It can handle ill-conditioned/rank-deficient problems
- It can handle nonconvex problems
- Inexactness is allowed and controlled with loose conditions
- The conditions are implementable (in fact, implemented)
- The algorithm is globally convergent
- It can handle problems with control and state constraints
- Numerical results are very encouraging so far
Future work and questions

What are we missing (to really solve PDE-constrained problems)?

- PDE-specific preconditioners
- Use of appropriate norms
- Mesh refinement, error estimators

What does it take to transform an algorithm for finite-dimensional optimization into one for solving infinite-dimensional problems?

- Can the finite-dimensional solver be a black-box?
- If not, to what extent do the outer and inner algorithms need to be coupled? (Do all components of the finite-dimensional solver need to be checked for their effect on the infinite-dimensional problem?)

What interesting problems may be solved with our approach?
References


