ICE05 Argonne, July 19, 2005

Nonlinear Optimization Solvers

Sven Leyffer leyffer@mcs.anl.gov

Mathematics & Computer Science Division,
Argonne National Laboratory

NONLINEAR OPTIMIZATION METHODS

Overview: Nonlinear Optimization Methods

Smooth nonlinear programming (NLP) problem

$$\begin{cases} \underset{x}{\text{minimize}} & f(x) & \text{objective} \\ \text{subject to} & c(x) = 0 & \text{constraints} \\ & x \geq 0 & \text{variables} \end{cases}$$

Solution methods:

1. active set methods: SQP & projected gradient
2. interior point methods

$\Rightarrow$ converge "near" solution x^* ...

Sequential Quadratic Programming (SQP)

General NLP

$$\underset{x}{\text{minimize}} \; f(x) \quad \text{subject to } c(x) = 0 \quad \& \quad x \geq 0$$

REPEAT

1. Solve QP for (s, y_{k+1}, z_{k+1})

$$\begin{cases} \underset{s}{\text{minimize}} & \nabla f_k^T s + \frac{1}{2} s^T H_k s \\ \text{subject to} & c_k + \nabla c_k^T s = 0 \qquad (\perp \; y_{k+1}) \\ & x_k + s \geq 0 \qquad (\perp \; z_{k+1} \geq 0) \end{cases}$$

2. Set $x_{k+1} = x_k + s$, & $k = k + 1$

UNTIL convergence

... "estimate" active set $\mathcal{A}_k := \{i : x_i = 0\}$

... quadratic convergence near x^* ... once $\mathcal{A}_*$ identified

Interior Point Methods (IPM)

General NLP

$$\underset{x}{\text{minimize}} \;\; f(x) \quad \text{subject to } c(x) = 0 \quad \& \quad x \geq 0$$

Perturbed $\mu > 0$ optimality conditions $(x, z \geq 0)$

$$F_\mu(x, y, z) = \left\{ \begin{array}{r} \nabla f(x) - \nabla c(x)^T y - z \\ c(x) \\ Xz - \mu e \end{array} \right\} = 0$$

- Primal-dual formulation
- Central path $\{x(\mu), y(\mu), z(\mu) \; : \; \mu > 0\}$
- Apply Newton's method for sequence $\mu \searrow 0$

Interior Point Methods (IPM)

REPEAT

1. Choose $\mu_k \searrow 0$
2. Solve $F_{\mu_k}(x, y, z) = 0$ to accuracy $\mathcal{O}(\mu)$

$$\begin{bmatrix} \nabla^2 \mathcal{L}_k & -\nabla c(x_k)^T & -I \\ \nabla c(x_k) & 0 & 0 \\ Z_k & 0 & X_k \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta z \end{pmatrix} = -F_{\mu_k}(x_k, y_k, z_k)$$

   where X_k diagonal matrix of x_k

3. Set $x_{k+1} = x_k + \Delta x$, ... & $k = k + 1$

UNTIL convergence

... SQP & IPM are variants of Newton's method

GLOBAL CONVERGENCE

Overview: Global Convergence

SQP & IPM converge quadratically "near" solution
... but diverge far from solution

convergence from remote starting points:

1. merit / penalty function to measure progress
2. enforce descent in merit function by ...
   - 2.1 line-search ... backtrack on SQP step:
     $x_k + \alpha \Delta x$ for $\alpha = 1, \frac{1}{2}, \ldots$
   - 2.2 restrict step with trust-region: $\|\Delta x\| \leq \rho_k$

... interfere as little as possible with SQP/IPM

Penalty & Merit Functions

Augmented Lagrangian (LANCELOT & MINOS)

$$\underset{x}{\text{minimize}} \; L(x, y_k, \rho_k) \; = \; f(x) - y_k^T c(x) + \tfrac{1}{2}\rho_k \|c(x)\|^2$$

As $y_k \longrightarrow y_*$:
- $x_k \longrightarrow x_*$ for $\rho_k > \bar{\rho}$
- No ill-conditioning, improves convergence rate

Exact penalty function: $\pi > 0$ penalty parameter

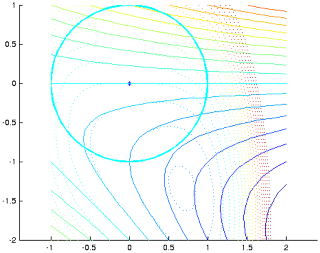
$$\underset{x}{\text{minimize}} \; \Phi(x, \pi) \; = \; f(x) + \pi \|c(x)\|$$

- equivalence of optimality $\Rightarrow$ exact for $\pi > \|y^*\|_D$
- nonsmooth $\Rightarrow$ Sℓ_1QP method

Trust Region Methods

Globalize SQP (IPM) by adding trust region, $\Delta^k > 0$

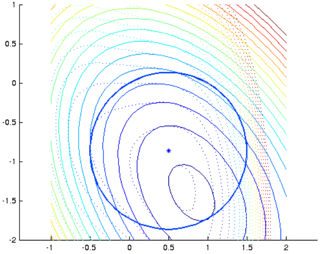
$$\begin{cases} \underset{s}{\text{minimize}} & \nabla f_k^T s + \frac{1}{2} s^T H_k s \\ \text{subject to} & c_k + A_k^T s = 0, \quad x_k + s \geq 0, \quad \|s\| \leq \Delta^k \end{cases}$$



Trust Region Methods

Globalize SQP (IPM) by adding trust region, $\Delta^k > 0$

$$\begin{cases} \underset{s}{\text{minimize}} & \nabla f_k^T s + \frac{1}{2} s^T H_k s \\ \text{subject to} & c_k + A_k^T s = 0, \quad x_k + s \geq 0, \quad \|s\| \leq \Delta^k \end{cases}$$



OPTIMIZATION SOFTWARE

Overview: Some Optimization Software

- Augmented Lagrangian
  - `LANCELOT`: bound constrained; trust-region
  - `MINOS`: linearly constrained; line-search
  - `PENNON`: line-search or trust-region
- Sequential quadratic programming
  - `FILTER`: trust-region; no penalty function
  - `KNITRO`: trust-region; SLP-EQP ... `options "alg=3";`
  - `SNOPT`: line-search; ℓ_1 exact penalty function
- Interior point methods
  - `KNITRO`: trust-region; SQP on barrier problem
  - `LOQO`: line-search; diagonal perturbation
  - `IPOPT`: line-search; no penalty function

Rule of Thumb for Solver Choice

Subjective rules of thumb ...

- trust-region more robust than line-search
- IPM faster than SQP ... but ...
  SQP more robust than IPM
- projected gradient is best for bound constrained optimization
- IPM may fail, if feasible set empty

Solver Options in AMPL

1. Setting solver options in AMPL:

```
option snopt_options "outlev=2 \
                      timing=1";
```

2. Useful solver options:

```
outlev     output level
timing     report solution times
maxiter    max. number of iterations
mu         initial barrier parameter
compl_frm  form of complements equation
```

3. Check with each solver for names, e.g.

```
knitro -=
```

... to see list of options (from `unix` prompt)

Life Cycle Savings

```
param T integer;            # number of periods
param beta, default 0.9; # utility param
param r    , default 0.2; # interest rate
param w{1..T} >= 0;        # wages in period t
var S{0..T} >= 0;          # savings in period t
var c{1..T} >= 0;          # consumption in period t

maximize utility:
         sum{t in 1..T} beta^t * (-exp(-c[t]));
subject to
   budget{t in 0..T-1}:
         S[t+1] = (1+r)*S[t] + w[t+1] - c[t+1];
   Cinit: S[0] = 0;
   Cend:  S[T] = 0;
```

Solver Output for SNOPT

```
Major Minors     Step   nObj Feasible  Optimal       Objective     nS
    0     14               1           3.0E-02 -2.7667695E+00    14   r
    1      2  6.8E-01      2           1.7E-02 -2.7087027E+00    13 n rl
    2      1  1.0E+00      3           1.1E-02 -2.6701545E+00    13 s
    8      1  1.0E+00      9          (2.3E-07)-2.6404137E+00    13

EXIT -- optimal solution found

Problem name                 problem
No. of iterations                 28   Objective value     -2.6404137441E+00
No. of major iterations            8   Linear objective     0.0000000000E+00
Penalty parameter          0.000E+00   Nonlinear objective -2.6404137441E+00
No. of calls to funobj            10   No. of calls to funcon              0
No. of superbasics                13   No. of basic nonlinears            20
No. of degenerate steps            5   Percentage                      17.86
Norm of x   (scaled)         6.4E+00   Norm of pi  (scaled)          1.0E+00
Norm of x                    3.6E+00   Norm of pi                    1.0E+00
Max Prim inf(scaled)       0 0.0E+00   Max Dual inf(scaled)       25 5.4E-07
Max Primal infeas          0 0.0E+00   Max Dual infeas            25 5.4E-07
```

Solver Output for LOQO

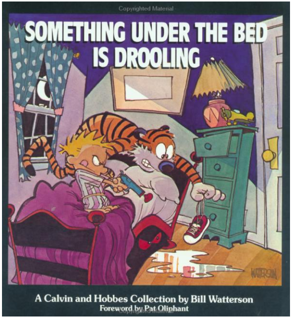
```
LOQO 6.06: outlev=2
variables: non-neg     49,  free        0,  bdd          0,  total      49
constraints: eq        25,  ineq        0,  ranged       0,  total      25
nonzeros:    A         73,  Q          25
------------------------------------------------------------------------------
     |         Primal          |          Dual            | Sig
Iter |  Obj Value      Infeas  |  Obj Value     Infeas    | Fig  Status  P   M
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
  1   -8.353892e+00   2.7e-01    -8.353892e+00   2.9e-01   60
nonzeros:    L         151,  arith_ops                880
  2   -4.629009e+00   1.3e-01    -1.193999e+01   1.5e-01
 10   -2.642555e+00   3.5e-06    -2.638877e+00   1.1e-04    3
 11   -2.640877e+00   4.7e-07    -2.640111e+00   2.4e-05    4   PF
 14   -2.640414e+00   6.5e-10    -2.640412e+00   1.2e-07    6   PF DF
 15   -2.640414e+00   3.6e-11    -2.640414e+00   1.2e-08    7   PF DF
----------------------
OPTIMAL SOLUTION FOUND; Finished call

LOQO 6.06: optimal solution (15 iterations, 15 evaluations)
primal objective -2.640413785
  dual objective -2.640413644
```

TROUBLESHOOTING

Something Under the Bed is Drooling

1. floating point (IEEE) exceptions
2. unbounded problems
   2.1 unbounded objective
   2.2 unbounded multipliers
3. (locally) inconsistent problems
4. suboptimal solutions



... identify problem & suggest remedies

Floating Point (IEEE) Exceptions

Bad example: minimize barrier function

```
param mu default 1;
var x{1..2} >= -10, <= 10;
var s;
minimize barrier: x[1]^2 + x[2]^2 - mu*log(s);
subject to
   cons: s = x[1] + x[2]^2 - 1;
```

... results in error message like

`Cannot evaluate objective at start`

... change initialization of `s`:

`var s := 1;` ... difficult, if IEEE during solve ...

Unbounded Objective

Penalized MPEC $\pi = 1$:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & x_1^2 + x_2^2 - 4x_1x_2 \quad + \pi x_1 x_2 \\ \text{subject to} & x_1, x_2 \geq 0 \end{array}$$

... unbounded below for all $\pi < 2$

Unbounded Multipliers for MPECs

A nasty MPEC example ... multipliers do not exist ...

```
var z{1..2} >= 0;
var z3;
minimize objf: z[1] + z[2] - z3;
subject to
   lin1:   -4 * z[1] + z3 <= 0;
   lin2:   -4 * z[2] + z3 <= 0;
   compl:  z[1]*z[2] <= 0;
```

... solution: `(z[1],z[2],z3) = (0,0,0)` with `objf = 0`

Unbounded Multipliers LOQO

```
LOQO 6.06: outlev=2
     |          Primal          |          Dual
Iter |  Obj Value     Infeas    |   Obj Value    Infeas
- - - - - - - - - - - - - - - - - - - - - - - - - - -
  1     1.000000e+00 0.0e+00     0.000000e+00 1.1e+00
  2     6.902180e-01 2.2e-01    -2.672676e-01 2.6e-01
  3     2.773222e-01 1.6e-01    -3.051049e-01 1.1e-01

292    -8.213292e-05 1.7e-09    -4.106638e-05 9.1e-07
293    -8.202525e-05 1.7e-09    -4.101255e-05 9.1e-07

294              nan    nan              nan      nan
500              nan    nan              nan      nan
```

nan = not a number ... desaster ... switch solver

Unbounded Multipliers FILTER

```
iter  |    rho      |   ||d||     |     f / hJ     |    ||c||/hJt
------+-------------+-------------+----------------+---------------
   0:0  10.0000       0.00000       1.0000000       0.0000000
   1:1  10.0000       1.00000       0.0000000       0.0000000
  24:1 0.156250     0.196695E-05-0.98347664E-06 0.24180657E-12
  25:1 0.156250     0.983477E-06-0.49173832E-06 0.60451644E-13

Norm of KKT residual................  0.471404521
max( |lam_i| * || a_i ||)...........  2.06155281

Largest modulus multiplier..........  2711469.25
```

- large multiplier of z[1]*z[2]<=0 $\Rightarrow \nexists$ multipliers
- NLP solvers converge linearly
- luckily this behaviour is very rare

Unbounded Complementarity Multiplier

Multiplier ξ of $X_1 x_2 \leq 0$ becomes large
$\Rightarrow \not\exists$ MPEC multipliers (B-stationary)

Remedy:

1. check residuals of constraints (if large then failed)
2. find degenerate indices $\mathcal{D} := \{i : |x_{1i}| + |x_{2i}| < \epsilon\}$
3. fix $x_{1i} = 0$ or $x_{2i} = 0$ for all $i \in \mathcal{D}$

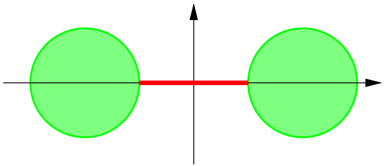
In previous example:

```
solve; fix z[1] := 0; solve;
```

Prove optimality: examine $2^{|\mathcal{D}|}$ combinations x_{1i} or $x_{2i} = 0$

Locally Inconsistent Problems

NLP may have no feasible point



feasible set: intersection of circles

Locally Inconsistent Problems

NLP may have no feasible point

```
var x{1..2} >= -1;

minimize objf:  -1000*x[2];

subject to
con1:  (x[1]+2)^2 + x[2]^2 <= 1;
con2:  (x[1]-2)^2 + x[2]^2 <= 1;
```

- not all solvers recognize this ...
- finding feasible point ⇔ global optimization

Locally Inconsistent Problems

LOQO

```
     |          Primal           |            Dual
Iter |  Obj Value      Infeas    |   Obj Value      Infeas
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
  1   -1.000000e+03   4.2e+00    -6.000000e+00   1.0e-00
[...]
500    2.312535e-04   7.9e-01     1.715213e+12   1.5e-01
LOQO 6.06: iteration limit
```

... fails to converge ... not useful for user

dual unbounded $\rightarrow \infty \Rightarrow$ primal infeasible

Locally Inconsistent Problems

FILTER

```
iter  |    rho    |    ||d||     |    f / hJ  |   ||c||/hJt
------+-----------+-------------+------------+------------
   0:0  10.0000    0.00000      -1000.0000    16.000000
   1:1  10.0000    2.00000      -1000.0000    8.0000000
[...]
   8:2  2.00000   0.320001E-02  7.9999693   0.10240052E-04
   9:2  2.00000   0.512000E-05  8.0000000   0.26214586E-10
filterSQP: Nonlinear constraints locally infeasible
```

... fast convergence to minimum infeasibility

... identify “blocking” constraints ... modify model/data

Locally Inconsistent Problems

Remedies for locally infeasible problems:

1. check your model: print constraints & residuals, e.g.
```
solve;
display _conname, _con.lb, _con.body, _con.ub;
display _varname, _var.lb, _var, _var.ub;
```
   ... look at violated and active constraints
2. try different nonlinear solvers (easy with AMPL)
3. build-up model from few constraints at a time
4. try different starting points ... global optimization

Inconsistent Problems: Sequential Approach

Example `heart6.mod`

⇒ `SNOPT: nonlinear infeasibilities minimized`
... after 549 iters

```
model heart6.mod;                  # load model
problem heart614: obj,             # dummy objective
        a, c, t, u, v, w,          # variables
        cons1, cons2, cons3,       # partial
        cons4;                     # ... constraints
solve heart614;                    # partial model
```

⇒ partial solution after 6 iterations ...

Inconsistent Problems: Sequential Approach

Example `heart6.mod`

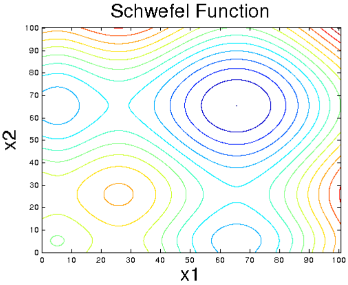
⇒ `SNOPT: nonlinear infeasibilities minimized`
... after 549 iters

```
# ... continue from prev. slide
problem heart616: obj,              # dummy objective
        a, c, t, u, v, w,           # variables
        cons1, cons2, cons3,        # all
        cons4, cons5, cons6;        # ... constraints
solve heart616;                     # full model
```

⇒ optimal solution after 6+20 iterations ...EXCELLENT

Suboptimal Solution & Multi-start

Problems can have many local mimimizers



default start point converges to local minimizer

Suboptimal Solution & Multi-start

Problems can have many local mimimizers

```
param pi := 3.1416;
param n integer, >= 0, default 2;
set N := 1..n;
var x{N} >= 0, <= 32*pi, := 1;
minimize objf:
- sum{i in N} x[i]*sin(sqrt(x[i]));
```

default start point converges to local minimizer

Suboptimal Solution & Multi-start

```
param nD := 5;        # discretization
set    D := 1..nD;
param  hD := 32*pi/(nD-1);
param optval{D,D};
model schwefel.mod;  # load model

for {i in D}{
   let x[1] := (i-1)*hD;
   for {j in D}{
      let x[2] := (j-1)*hD;
      solve;
      let optval[i,j] := objf;
   }; # end for
}; # end for
```

Suboptimal Solution & Multi-start

```
display optval;
optval [*,*]
:     1         2          3          4          5    :=
1     0         24.003    -36.29     -50.927     56.909
2    24.003    -7.8906    -67.580    -67.580    -67.580
3   -36.29     -67.5803   -127.27    -127.27    -127.27
4   -50.927    -67.5803   -127.27    -127.27    -127.27
5    56.909    -67.5803   -127.27    -127.27    -127.27
;
```

... there exist better multi-start procedures

CONCLUSIONS

Conclusions

Nonlinear optimization much harder than LP:

- there is no “best” solver
- best solver is problem dependent
- learn a little about solvers/methods
  $\Rightarrow$ help with troubleshooting
- multistart & model build-up in AMPL
- please write good models ...