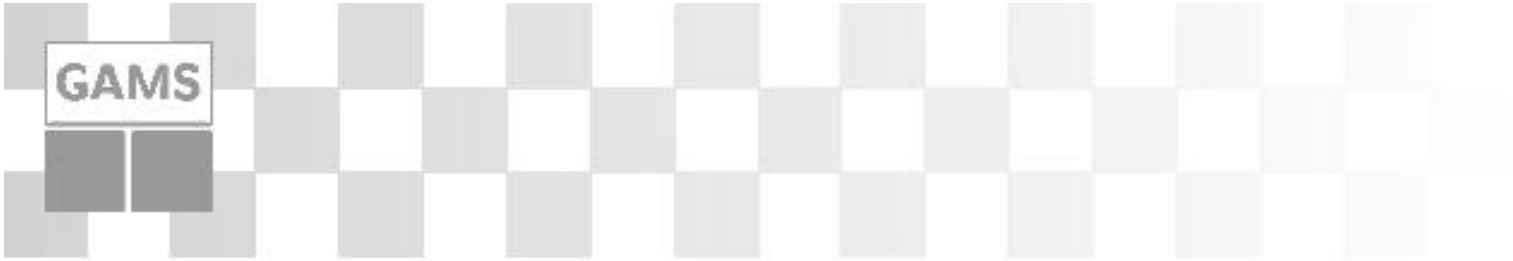


PATHNLP: Solving NLPs as complementarity problems

Steven Dirkse

GAMS Development Corporation

Washington DC

Overview

- PATH - the solver for MCP
- Difficult NLPs, computational failures
- KKT conditions of NLP == MCP
- PATHNLP
- Implementation & computational results

MCP and Normal Equations

- MCP:

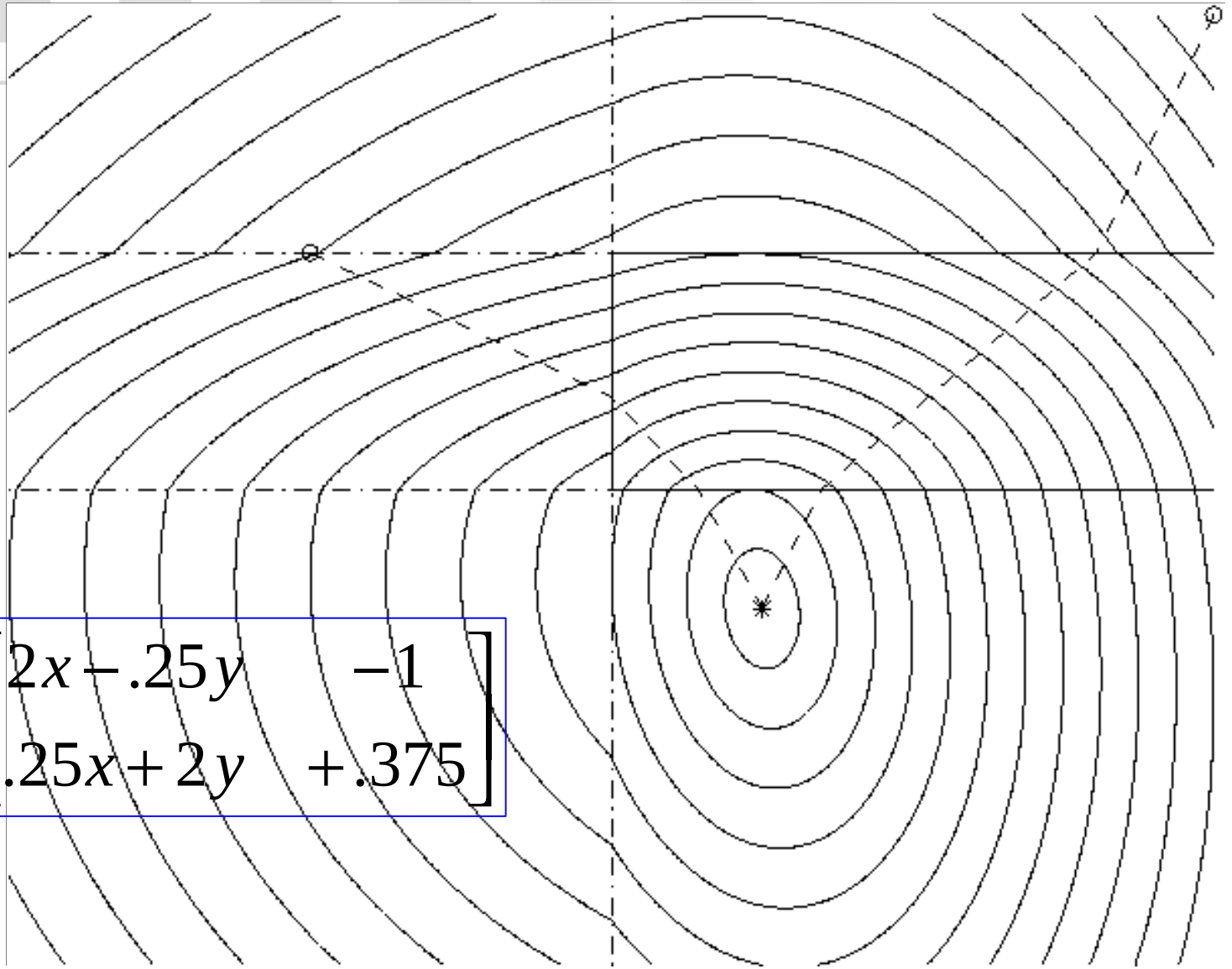
$$F(z) \quad \perp \quad z \in [L, U]$$

- Normal Equations:

$$F(p(x)) + x - p(x) = 0,$$

where $p(x) =$ projection onto $[L, U]$

A Normal Map



$$F(x,y)=\begin{bmatrix} 2x-.25y & -1 \\ .25x+2y & +.375 \end{bmatrix}$$

Basis Crashing in PATH

- Poor initial basis implies expensive path computation with many pivots
- Crash technique:
  - Compute direction from current point
  - Take full step, then project onto bounds
  - Damp with Armijo linesearch

Quadratic Convergence in PATH

- Proof of quadratic convergence requires
  - uniformity, invertibility of linearization
  - usual Newton condition on the linearization error
- In practice, quadratic convergence is usually observed
- Near the solution, PATH behaves as a smooth Newton solver

Gambling Model

- max $\sum_i c_i * \log(f(x_i, y))$
- s.t. $y = \sum_i x_i$, $y \leq W$

- size of model parameterized by N:

N	6	7	8	9	10	11	14
n	98	509	1261	3186	4151	8885	27328

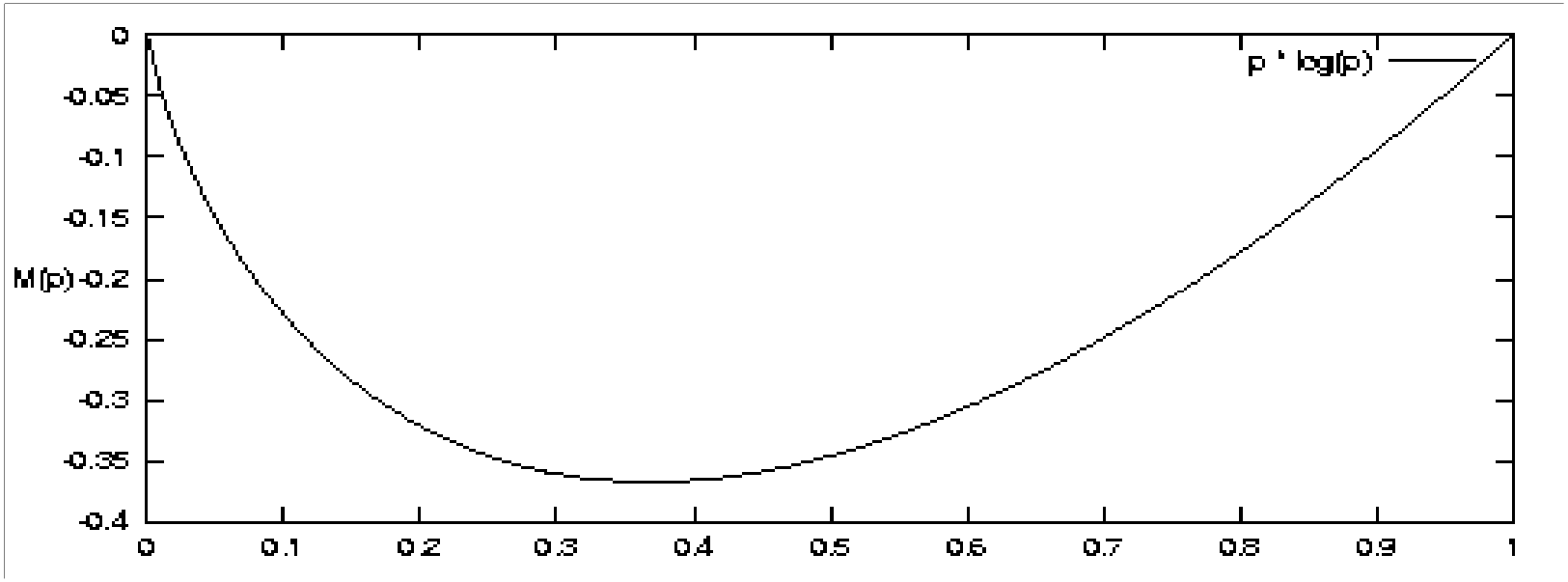
Off-the-shelf computation

- GAMS NLP codes (gambling model)

N	6	7	8	9	10	11	12
n	98	509	1261	3186	4151	8885	16671
CONOPT	0	18	152	time			
MINOS	0	2	9	66	93	309	5506
SNOPT	0	37	980	20521			

Entropy Maximization Models

- Maximize entropy M or cross-entropy CE
  - $M(p) = -S\ p_i * \log(p_i)$
  - $CE(p) = -S\ p_i * \log(p_{i/}\ q_i)$



Example: Data Reconciliation

- Given:
  - Sets of households H, labor segments S
  - A(H,S) - participation in labor force
  - w(H) - statistical weights of households
  - y(S) - observed structure of labor force
- Problem: Aw != y
- Normalize w to get prior distribution q
- Max CE(p) s.t. Ap = y_n

Example: Data Reconciliation

- MaxEnt model from Denmark (DIAFE)
- Model is quite large
  - 277,017 variables
  - 6805 constraints
  - 3,587,265 nonzeros
- Client asks, “What machine must I buy to solve this model?”

First-Order Conditions for NLP

- NLP:

$$\begin{array}{lll} \min & f(x) & \\ \text{s.t.} & g(x) \geq 0, & x \in [L,U] \end{array}$$

- First-Order Conditions (KKT):

$$\begin{array}{lcl} \nabla f(x) \;\text{-}\; \nabla g(x) \cdot u & \perp & x \in [L,U] \\ g(x) \geq 0 & \perp & u \in [0,\infty] \end{array}$$

NLP2MCP (Ferris & Horn)

- GAMS source translation tool
- Result is source of MCP model (symmetric)
- Preserves GAMS structure as much as possible (adding asymmetry possible)
- Not completely robust wrt. GAMS syntax
- Used by several economists to solve large MaxEnt models via MCP

PATHNLP Algorithm

- Internally reformulate NLP via KKT conditions
  - Requires 1st derivatives
- Solve KKT conditions via PATH
  - Requires 2nd derivatives
  - Convexity required for convergence proof
- Extract NLP solution from KKT solution

G2DLIB (Drud, ...)

- G2DLIB = 2nd deriv capability, & more
- Current capability
  - 1st derivative (backwards & forwards)
  - 2nd derivative (forwards)
  - interval analysis for functions, derivatives
- Current/future work
  - gradient-vector, Hessian-vector products
  - forward-reverse mode Hessian
  - Preprocessing, evaluation reordering

Performance (gambling)

N	7	8	9	10	11	12	13	14
n	509	1261	3186	4151	8885	16671	21856	27328
CONOPT	18	152	time					
MINOS	2	9	66	93	309	5506		
SNOPT	37	980	20521					
PATHNLP	1	4	34	77	579	558	1089	1689

gambling model, on jfk.gams.com

Solution times (MaxEnt)

Solver / model	CONOPT2	MINOS	SNOPT	PATHNLP
y1996	179	698	10800	18
y1997	154	677	10800	15
y1998	214	340	10800	16
y1999	275	482	10800	16

y1999 model, on jfk.gams.com

Performance (2nd deriv)

- How costly is obtaining Hessian information?
  - MaxEnt models: computed quickly
  - Not the case for gambling model

% time in Hessian eval

N	9	10	11	12	13	14
%	15%	11%	7%	64%	66%	65%

Performance (modlib)

- GAMS modlib: 99 NLP solves
  - 96 successes
  - 2 failures (poor memory estimate)
  - 1 failure (non-convex QP)
- COPS models in modlib
  - Mixed success
  - Lack of objective guidance apparent
  - Feast or famine

PATHNLP: Pros & Cons

- Pros:
  - Excellent “warm/hot start” capability
  - No superbasics limit
  - Quadratic convergence
- Cons:
  - Potential lack of robustness
  - Diagnosis of infeasible or unbounded models

Infeasible Models

- Introduce Phase I NLP:

$$\min \quad < s, s >$$

$$\text{s.t.} \quad g(x) + s \geq 0, \quad x \in [L, U]$$

- Solution determines (local) infeasibility
- Feasibility not necessarily maintained in subsequent solve of the original MCP
- Issue: run Phase I first, or only on failure of original model?

Improving robustness

- PATH may fail to solve KTT conditions, or find a KKT point for a maximizer.
- Phase I determines (local) infeasibility
- If feasible, we can resolve KKT starting from Phase I solution
- Composite merit function (Krung & Ferris)
  - Makes use of NLP objective
  - Favors minimizers over maximizers

SBB: PATH vs. CONOPT

Objective & Speed Improvements

speed / objective	slower	same	better	
poorer	61	7	2	70
same	19	18	4	41
better	4	4	4	12
	84	29	10	123

SBB: PATH vs. CONOPT

Objective & Speed Improvements

speed / objective	slower	same	better	
poorer	3	1	1	5
same	18	18	3	39
better	4	3		7
	25	22	4	51

Conclusions

- Using 2nd order information is important and now available in GAMS
- Important classes of NLP models now solvable (option NLP=PATHNLP)
- SBB robustness improved by addition of PATHNLP
- Open research: improvements in efficiency, robustness