Path-Following Algorithm for Posynomial Geometric Programming

This Hypercard stack was prepared by:
Dennis L. Bricker,
Dept. of Industrial Engineering,
University of Iowa,
Iowa City, Iowa 52242
e-mail: dbricker@icaen.uiowa.edu

Consider the linearly-constrained convex programming problem

$$\Phi^* = \text{Minimum } F(x)$$
$$\text{subject to } Ax=b$$
$$x \geq 0$$

where F is a convex function.

Barrier function:

$$\Phi(\zeta) = \text{Minimum } F(x) - \zeta \sum_{j=1}^{n} \ln x_j$$
$$\text{subject to } Ax=b, \quad x \geq 0$$

as $x \longrightarrow 0, \quad -\zeta \ln(x) \longrightarrow \infty$

so the minimizer $x(\zeta)$ of $\Phi(\zeta)$ is positive!

Lagrangian function:

$$L(x,y,\zeta) = F(x) - \zeta \sum_{j=1}^{n} x_j - y^T (Ax - b)$$

Optimality conditions for $\Phi(\zeta)$:

$$\frac{\partial L(x,y,\zeta)}{\partial x_j} \geq 0, \quad j=1,\ldots n$$

$$\frac{\partial L(x,y,\zeta)}{\partial y_i} = 0, \quad i=1,\ldots m$$

$$x_j \left(\frac{\partial L(x,y,\zeta)}{\partial x_j} \right) = 0$$

complementary slackness conditions

$$\frac{\partial L(x,y,\zeta)}{\partial x_j} \geq 0, \quad j=1,\ldots n$$

$$\Rightarrow \quad \frac{\partial}{\partial x_j} L(x,y,\zeta) = \frac{\partial F}{\partial x_j} - \frac{\zeta}{x_j} - \sum_{i=1}^{m} y_i a_{ij} \geq 0$$

$$\frac{\partial L(x,y,\zeta)}{\partial y_i} = 0, \quad i=1,\ldots m$$

$$\Rightarrow \quad Ax = b$$

Notation

$$e^T \equiv (1, 1, 1, \ldots 1)$$

$$X \equiv \begin{bmatrix} x_1 & 0 & 0 & \cdots & 0 \\ 0 & x_2 & 0 & \cdots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & & x_n \end{bmatrix} \Longrightarrow X^{-1} = \begin{bmatrix} \frac{1}{x_1} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{x_2} & 0 & \cdots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & & \frac{1}{x_n} \end{bmatrix}$$

$$\frac{\partial L(x,y,\zeta)}{\partial x_j} \geq 0, \quad j=1,\ldots n$$

$$\Rightarrow \qquad \nabla F(x) - A^T y - \zeta X^{-1} e \geq 0$$

$$x \left(\nabla F(x) - A^T y - \zeta X^{-1} e \right) = 0$$

complementary slackness conditions

The minimizer is positive, i.e., $x(\zeta) > 0$

so that it must satisfy $\nabla F(x) - A^T y - \zeta X^{-1} e = 0$

together with $Ax = b$

Optimality conditions for $\Phi(\zeta)$ are

$$\begin{cases} \nabla F(x) - A^T y - \zeta X^{-1} e = 0 \\ Ax = b \end{cases}$$

Define

$$s \equiv \nabla F(x) - yA^T$$

so that the first condition is

$$s - \zeta X^{-1} e = 0$$

$$\Rightarrow \quad \begin{cases} X s = \zeta e \\ A x = b \end{cases}$$

Optimality Conditions

$$X\,s = \zeta\,e \quad \Rightarrow \quad x_j\,s_j = \zeta \quad \forall \quad j=1,\ldots n$$

$$x_j\,s_j = 0$$

Complementary
Slackness condition
for the problem Φ^*

i.e.,

ζ is the violation in each complementary slackness condition of the problem Φ^*

$$\begin{cases} X^k \, \mathbf{s}^k = \zeta^k \, \mathbf{e} \\ A \, \mathbf{x}^k = \mathbf{b} \end{cases}$$

$$\mathbf{s}^k = \nabla F(\mathbf{x}^k) - A^T \mathbf{y}^k$$

We want to use an iteration of the Newton-Raphson method to solve the nonlinear system

$$\begin{cases} X \, \mathbf{s} = \hat{\zeta} \, \mathbf{e} & \hat{\zeta} < \zeta^k \\ A \, \mathbf{x} = \mathbf{b} \end{cases}$$

$$\mathbf{s} = \nabla F(\mathbf{x}) - A^T \mathbf{y}$$

$$X\,s \approx X^k\,s^k + X^k\,\Delta s + S^k\,\Delta x$$

$$\Delta s \approx \nabla^2 F(x^k)\,\Delta x - A^T\,\Delta y$$

$$X\,s \approx X^k\,s^k + X^k\left(\nabla^2 F(x^k)\,\Delta x - A^T\,\Delta y\right) + S^k\,\Delta x$$

Setting $X\,s = \hat{\zeta}\,e$ gives

$$X^k\,s^k + X^k\left(\nabla^2 F(x^k)\,\Delta x - A^T\,\Delta y\right) + S^k\,\Delta x = \widehat{\zeta}\,e$$

$$X^k s^k + X^k \left(\nabla^2 F(x^k) \Delta x - A^T \Delta y \right) + S^k \Delta x = \hat{\zeta} e$$

Since $x^k > 0$, X^k is nonsingular, and

$$s^k + \nabla^2 F(x^k) \Delta x - A^T \Delta y + (X^k)^{-1} S^k \Delta x = \hat{\zeta} (X^k)^{-1} e$$

Newton-Raphson step is found by solving the *LINEAR* system:

$$\begin{cases} \left[\nabla^2 F(x^k) + (X^k)^{-1} S^k\right] \Delta x - A^T \Delta y = \hat{\zeta} (X^k)^{-1} e - s^k \\ \\ A \Delta x = 0 \end{cases}$$

and

$$\begin{cases} x^{k+1} = x^k + \Delta x \\ y^{k+1} = y^k + \Delta y \\ s^{k+1} = \nabla F(x^{k+1}) - A^T y^{k+1} \\ z^{k+1} = \frac{1}{n} x^{k+1} s^{k+1} \end{cases}$$

If we reduce the barrier parameter by

$$\widehat{\zeta} = \omega \ \zeta$$

for some factor ω, $0 < \omega < 1$
then we can perform another Newton-Raphson step to solve (approximately) the new nonlinear system of equations:

$$\begin{cases} X\ \mathbf{s} = \widehat{\zeta}\ \mathbf{e} \\ A\ \mathbf{x} = \mathbf{b} \end{cases}$$

Geometric Programming Primal Problem

$$\begin{cases} \text{Minimize} \quad g_0(t) \\ \text{subject to} \quad g_k(t) \le 1, \quad k=1,\cdots K \\ t_i > 0 \ , \ i=1,2,\cdots N \end{cases}$$

$$\text{where} \quad g_k(t) = \sum_{j\in[k]} c_j \prod_{i=1}^{N} t_i^{a_{ij}}$$

posynomial

$$c_j > 0$$

$$\bigcup_k [k] = \{1,2,\cdots N\} \quad \& \quad [k'] \cap [k''] = \varnothing \quad \text{for } k' \neq k''$$

Application to Geometric Programming Dual Problem

The GP dual problem is linearly-constrained, and (if the negative of the log of the objective is minimized) has a convex objective function.

DGP: Maximize $v(\delta,\lambda) = \prod_{k=0}^{K} \left\{ \lambda_k^{\lambda_k} \prod_{i\in[k]} \left(\frac{c_i}{\delta_i} \right)^{\delta_i} \right\}$

subject to

$$\sum_{i\in[k]} \delta_i = \lambda_k \ , \ k=0,1,\cdots K$$

Geometric Programming Dual Problem

$$\sum_{i=1}^{N} a_{ij}\, \delta_i = 0, \quad j=1,\cdots M$$

$$\lambda_0 = 1$$

$$\delta_i \geq 0,\ \lambda_k \geq 0 \quad \forall i,k$$

Note: $\bigcup_k [k] = \{1,2,\cdots N\}$ & $[k'] \cap [k''] = \varnothing$ for $k' \neq k''$

$$\text{Maximize} \quad v(\delta,\lambda) = \prod_{k=0}^{K} \left\{ \lambda_k^{\lambda_k} \prod_{i \in [k]} \left(\frac{c_i}{\delta_i} \right)^{\delta_i} \right\}$$

is equivalent to

$$\text{Max} \quad \ln v(\delta,\lambda) = \sum_{i=1}^{N} \left\{ \delta_i \ln c_i - \delta_i \ln \delta_i \right\} + \sum_{k=0}^{K} \lambda_k \ln \lambda_k$$

or

$$\text{Min} \quad -\ln v(\delta,\lambda) = \sum_{i=1}^{N} \left\{ \delta_i \ln \delta_i - \delta_i \ln c_i \right\} - \sum_{k=0}^{K} \lambda_k \ln \lambda_k$$

$$\text{Min } -\ln v(\delta,\lambda) = \sum_{i=1}^{N} \{\delta_i \ln \delta_i - \delta_i \ln c_i\} - \sum_{k=0}^{K} \lambda_k \ln \lambda_k$$

The above objective is convex if we make the substitution $\sum_{i\in[k]} \delta_i = \lambda_k$

$$\text{Min } -\ln V(\delta) = \sum_{i=1}^{N} \{\delta_i \ln \delta_i - \delta_i \ln c_i\} - \sum_{k=0}^{K} \left[\sum_{i\in[k]} \delta_i\right] \ln \left[\sum_{i\in[k]} \delta_i\right]$$

$$\text{DGP}': \ \text{Min} \ \sum_{i=1}^{N} \{\delta_i \ln \delta_i - \delta_i \ln c_i\} - \sum_{k=0}^{K} \left[\sum_{i\in[k]} \delta_i\right] \ln \left[\sum_{i\in[k]} \delta_i\right]$$

subject to

$$\sum_{i\in[k]} \delta_i = 1 \qquad \textit{normality}$$

Geometric Programming Dual Problem

$$\sum_{i=1}^{N} a_{ij}\, \delta_i = 0, \ \ j=1, \cdots M \qquad \textit{orthogonality}$$

$$\delta_i \geq 0, \quad \forall i$$

The algorithm which we have described was implemented in an experimental APL code PFAGP1.

$$\begin{aligned} x &\longleftrightarrow \delta \\ y &\longleftrightarrow \ln\, t \\ F(x) &\longleftrightarrow -\ln V(\delta) \end{aligned}$$

The primal GP variables (t) are obtained by exponentiating y

Obtaining the starting solution (x^0, s^0, y^0)

$$\begin{cases} X^0 \, \mathbf{s}^0 = \zeta^0 \, \mathbf{e} \\ A \, \mathbf{x}^0 = \mathbf{b} \qquad\qquad \mathbf{s}^0 = \nabla F(\mathbf{x}^0) - A^T \mathbf{y}^0 \end{cases}$$

is accomplished by arbitrarily choosing x & y,
& adding

an artificial variable x_{N+1} with large cost,

a bounding constraint $\sum_{i=1}^{N+1} x_i \le U$ with large U

setting $\zeta^0 = \frac{1}{N+1} \sum_{i=1}^{N+1} x_i^0 s_i^0$

Computational Experience

Beck & Ecker's Problem #10 will be used to illustrate the behavior of this algorithm

- # variables = 7
- # constraints = 4
- # terms = 20
- # "degrees of difficulty" = 12

There are 6 different variations, in which the exponent α in one term is varied.

As the exponent α is varied from $-1/4$ to $+1$, the first & last constraints change from being both slack to both active.

The GP dual problem is rather badly conditioned. For example, in the case $\alpha=+1$, the optimal values of three of the positive dual variables are less than 0.001, and most dual-based GP algorithms experience difficulty in determining the optimal primal variables.

B&E #10A

iteration	z	ω	z'/z
1	2.278E0	94.05	102.17
2	2.128E0	84.88	93.41
3	1.734E0	76.60	81.51
4	1.266E0	69.14	72.96
5	8.326E-1	62.39	65.79
6	4.953E-1	56.31	59.49
7	2.679E-1	50.82	54.08
8	1.331E-1	45.87	49.68
9	5.998E-2	41.39	45.08
10	2.461E-2	37.36	41.03
11	9.178E-3	33.72	37.29
12	3.111E-3	30.43	33.89
13	9.550E-4	27.46	30.70
14	2.643E-4	24.78	27.68
15	6.585E-5	22.37	24.91
16	1.460E-5	20.19	22.18
17	2.846E-6	18.22	19.48
18	4.939E-7	16.44	17.36
19	7.718E-8	14.84	15.63
20	1.088E-8	13.39	14.10
21	1.384E-9	12.09	12.72
⋮	⋮	⋮	⋮

***Warning: Δx results in x+Δx<0 at iteration 1
(x[2] = 0 at x + 0.53752306×Δx)
Step used is 0.48377075×Δx

B&E #10A

iteration	z	ω	z'/z
⋮	⋮	⋮	⋮
22	1.589E-10	10.91	11.48
23	1.647E-11	9.84	10.36
24	3.208E-12	14.41	19.48

***Deleting artificial variable, which is 5.4356152E-14 at iteration 25

25	5.435E-13	16.10	16.94
26	8.313E-14	14.53	15.30
27	1.147E-14	13.11	13.79
28	1.420E-15	11.83	12.38

Converged in iteration # 27

B&E #10A

i	x	s	xs
1	5.5605774E-1	2.6645353E-15	1.4816354E-15
2	4.4336474E-1	2.6645353E-15	1.1813610E-15
3	2.3683684E-4	6.0289551E-12	1.4278787E-15
4	3.4068164E-4	4.1922021E-12	1.4282063E-15
5	6.2970015E-16	2.2683319E0	1.4283689E-15
6	6.1879025E-15	2.3083249E-1	1.4283689E-15
7	2.6972464E-15	5.2956560E-1	1.4283689E-15
8	6.1263254E-1	1.9984014E-15	1.2242858E-15
9	1.3796628E0	1.1934898E-15	1.6466135E-15
10	8.7790268E-2	1.5987212E-14	1.4035216E-15
11	1.0723557E0	1.1102230E-15	1.1905540E-15
12	4.3822934E-1	3.1086245E-15	1.3622904E-15
13	7.9114982E-1	1.9984014E-15	1.5810349E-15
14	2.2201890E-1	7.3274720E-15	1.6268373E-15
15	3.3192332E-16	4.3033099E0	1.4283689E-15
16	2.3568874E-15	6.0604037E-1	1.4283689E-15
17	5.1921119E-16	2.7510365E0	1.4283689E-15
18	1.2589491E-15	1.1345724E0	1.4283689E-15

Dual GP objective = 7.5009522 Objective function: 1809.7648
= log 1809.7648) (gap = -6.0878619E-8 i.e., -3.3638968E-9%)

B&E #10A

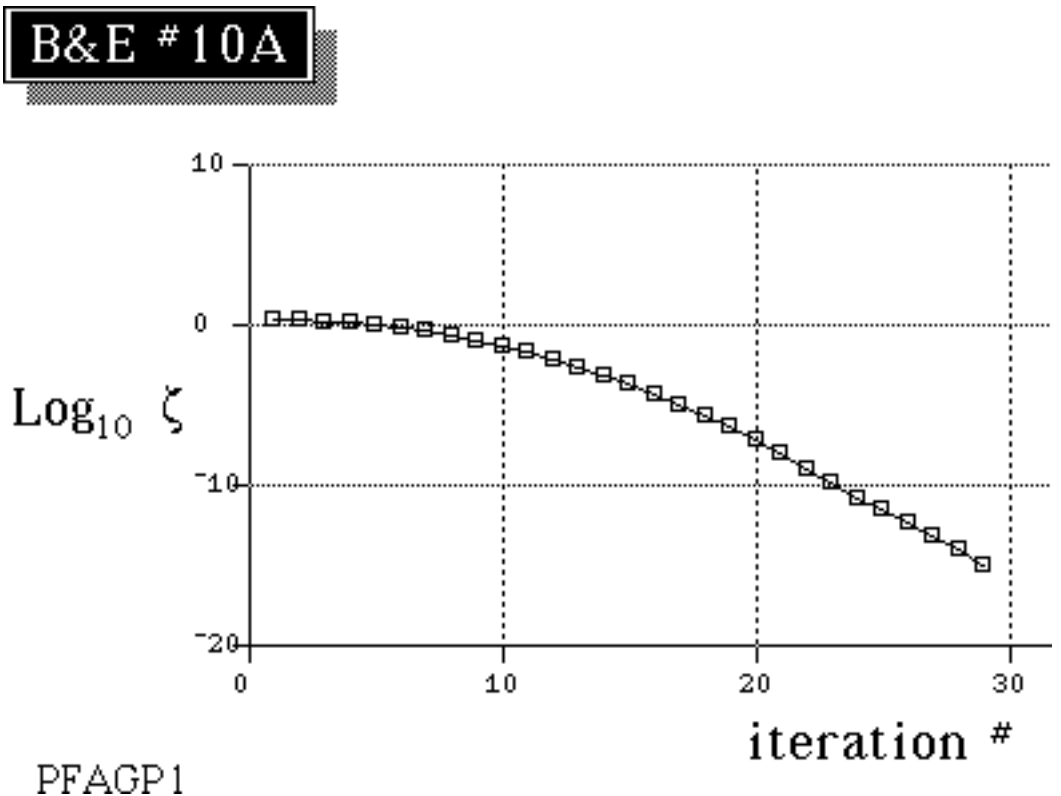
Dual variables (of the dual GP problem) = y:

i	y[i]	i	t[i]
1	-7.5009522E0	1	2.8561586E0
2	1.0494776E0	2	6.1082303E-1
3	-4.9294800E-1	3	2.1508126E0
4	7.6584571E-1	4	4.7128737E0
5	1.5502979E0	5	9.9948754E-1
6	-5.1259050E-4	6	1.3475075E0
7	2.9825659E-1	7	3.1652767E-2
8	-3.4529297E0		

z = 1.4196001E-15

Constraints:

Posy	Value	Infeasibility	Lambda
2	0.6900658082	0.0000000E0	9.5148491E-15
3	1.0000000000	0.0000000E0	2.0800857E0
4	1.0000000000	0.0000000E0	2.5237537E0
5	0.3868776663	0.0000000E0	4.4669711E-15


B&E #10A
10
0
-10
-20
0
10
20
30
$\mathrm{Log}_{10}\ \zeta$
iteration #
PFAGP1

B&E #10F

i	x	s	xs
1	7.2559669E-2	1.9539925E-14	1.4178105E-15
2	9.1462207E-1	4.4408921E-16	4.0617379E-16
3	1.2262304E-3	1.1759482E-12	1.4419835E-15
4	1.1592026E-2	1.2345680E-13	1.4311144E-15
5	1.5132990E-3	9.5257136E-13	1.4415253E-15
6	7.1002941E-2	2.0733415E-14	1.4721334E-15
7	1.1267145E-2	1.2789769E-13	1.4410418E-15
8	2.7919468E-1	5.5511151E-15	1.5498418E-15
9	9.0549382E-1	1.3600232E-15	1.2314926E-15
10	1.3509246E-1	1.0214052E-14	1.3798414E-15
11	1.1993180E0	8.8817842E-16	1.0652084E-15
12	6.1871705E-1	2.2204460E-15	1.3738278E-15
13	4.8434933E-1	2.8865799E-15	1.3981130E-15
14	1.5302901E-1	9.3258734E-15	1.4271292E-15
15	2.3148566E-4	6.2279071E-12	1.4416712E-15
16	3.3710643E-1	4.4408921E-15	1.4970533E-15
17	2.6793648E-4	5.3823612E-12	1.4421309E-15
18	1.0659804E-2	1.3544721E-13	1.4438406E-15

Dual GP objective = 6.0212145
= log 412.07877)

z = 1.3501074E-15
Objective function: 412.07877
(gap = -1.1523923E-8 i.e., -2.7965341E-9%)

PFAGP1

B&E #10F

Dual variables (of the dual GP problem) = y:

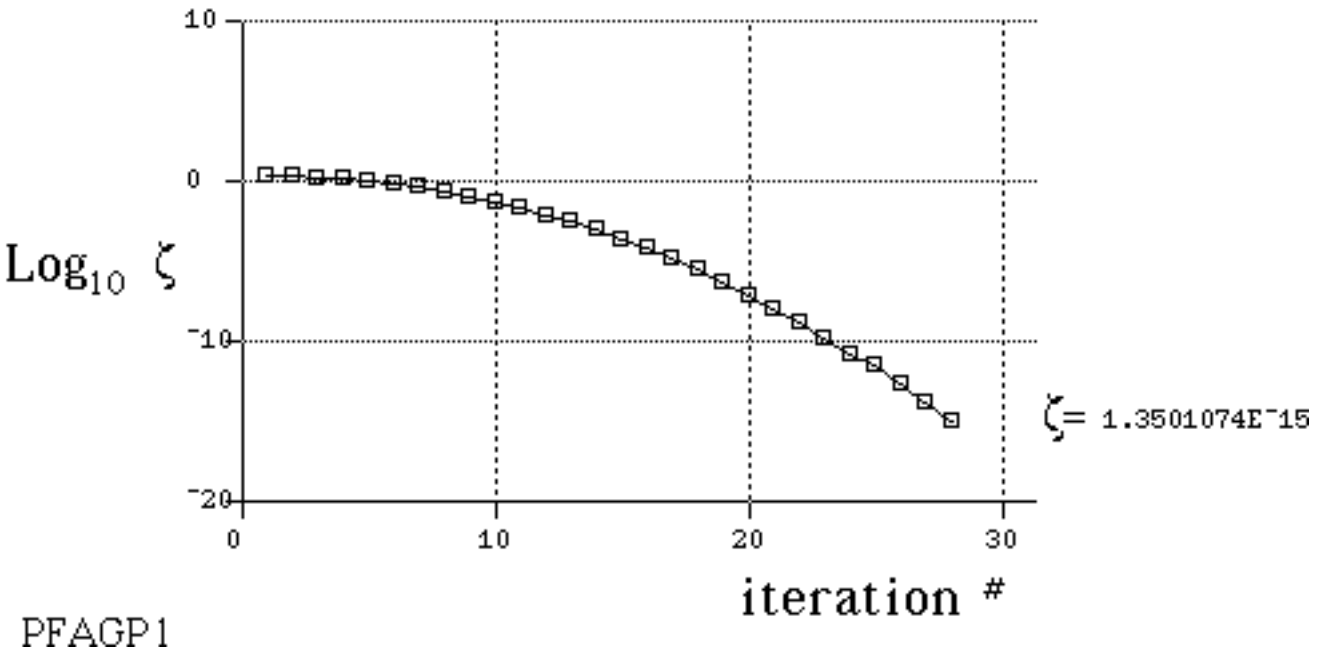
i	y[i]
1	-6.0212145E0
2	1.5098922E0
3	-1.8874188E-1
4	1.0374129E0
5	1.0785433E0
6	-4.4331639E-1
7	-2.7789153E-1
8	-3.5941118E0

i	t[i]
1	4.5262428E0
2	8.2800020E-1
3	2.8219069E0
4	2.9403930E0
5	6.4190409E-1
6	7.5737897E-1
7	2.7485085E-2

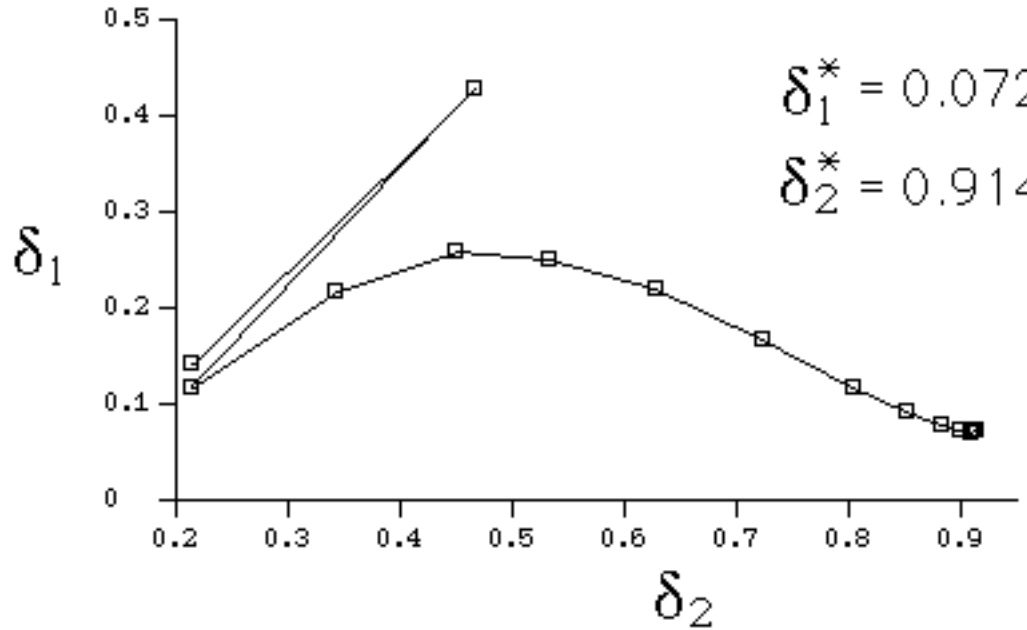
Constraints:

Posy	Value	Infeasibility	Lambda
2	1.0000000000	0.0000000E0	8.3783385E-2
3	1.0000000000	0.0000000E0	1.3197810E0
4	1.0000000000	0.0000000E0	2.4554134E0
5	1.0000000000	0.0000000E0	3.4826565E-1

B&E #10F



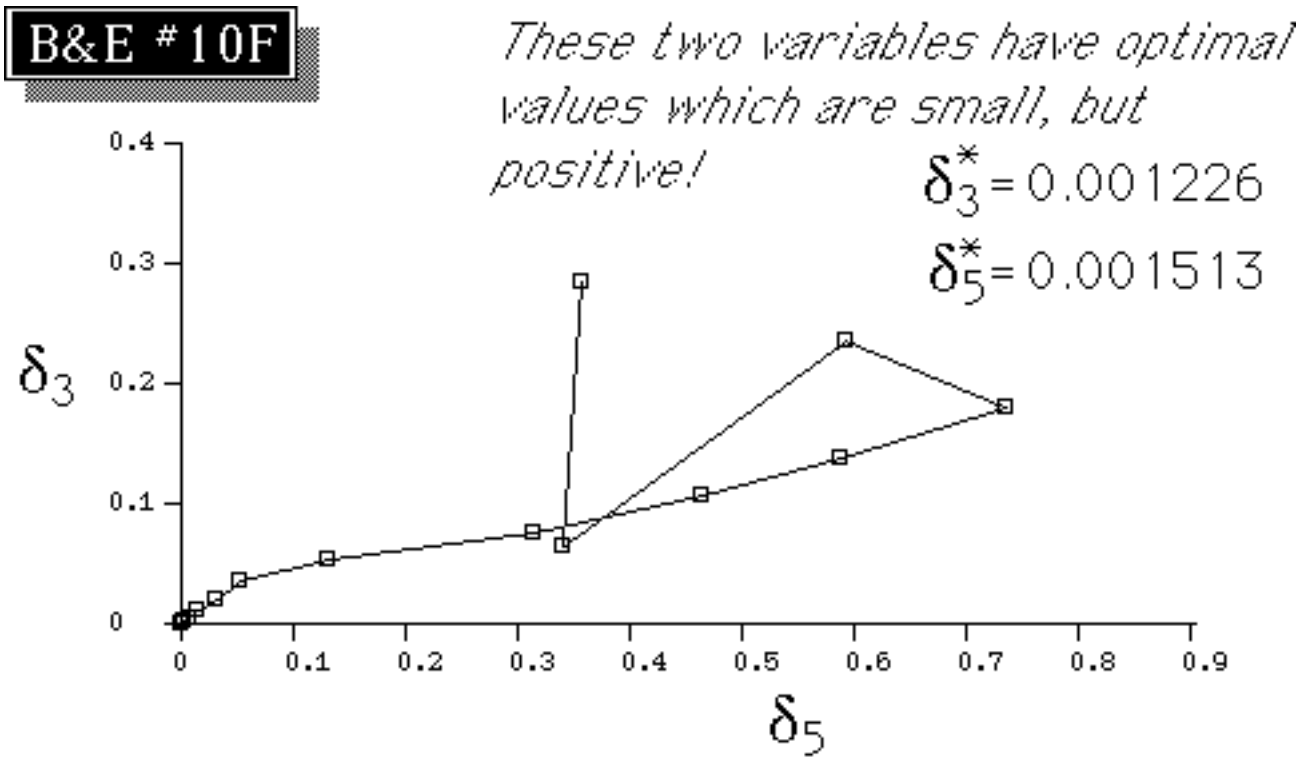
B&E #10F



$\delta_1^* = 0.072559669$

$\delta_2^* = 0.91462207$

PFAGP1


B&E #10F
These two variables have optimal values which are small, but positive!
$\delta_3^* = 0.001226$
$\delta_5^* = 0.001513$
δ_3
δ_5
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

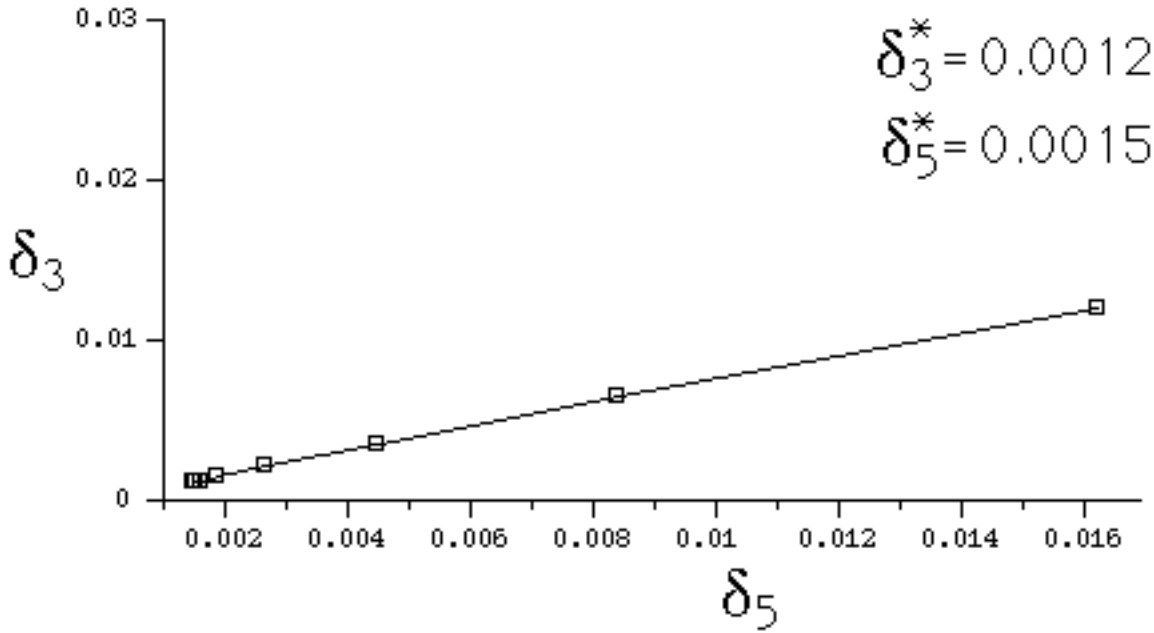
PFAGP1

B&E #10F

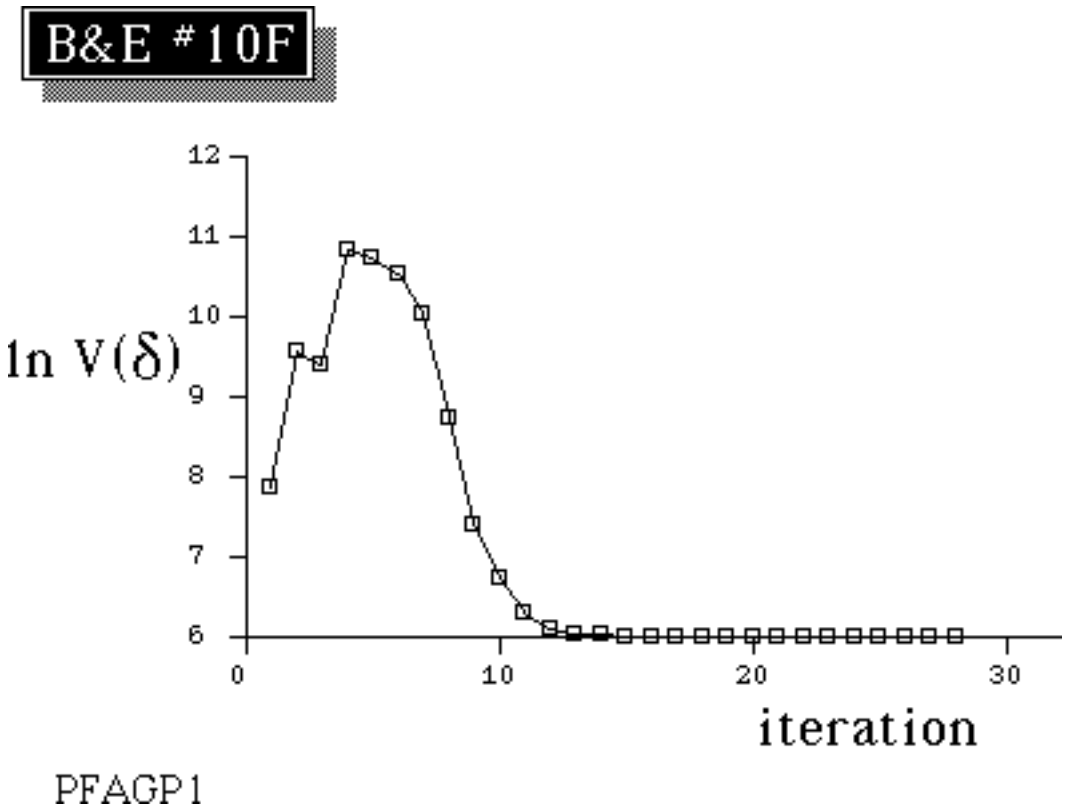
...looking at the final iterates

$\delta_3^* = 0.001226$

$\delta_5^* = 0.001513$



PFAGP1


B&E #10F
12
11
10
9
8
7
6
ln V(δ)
0
10
20
30
iteration
PFAGP1

Dual-based GP algorithms generally have difficulty with problems such as the following:

Minimize $g_0(t) = t_1 t_2 + t_1^{-1} t_2^{-1}$

subject to

$$g_1(t) = \frac{1}{4} t_1^{0.5} + t_2 \leq 1$$

$$t_1 > 0,\ t_2 > 0$$

The dual problem has a unique solution

$$\delta_1^* = \delta_2^* = 1/2, \lambda_1 = \delta_3^* = \delta_4^* = 0$$

The primal solution, which is not unique, cannot be determined by solving the equations

$$c_j \prod_i x_i^{a_{ij}} = \frac{\delta_j}{\lambda_k} \text{ where } j \in [k]$$

i	x	s	xs
1	5.0000000E-1	1.8651747E-14	9.3258734E-15
2	5.0000000E-1	1.8429702E-14	9.2148511E-15
3	4.2349134E-14	2.3035641E-1	9.7553946E-15
4	2.1108084E-14	4.0233010E-1	8.4924176E-15
5	9.0000000E0	1.0144972E-15	9.1304747E-15
6	1.0227926E-15	8.9478720E0	9.1518177E-15

(Artificial variable: 1.0227926E-15)
(Slack variable: 9)
z = 9.1784715E-15

Dual GP objective = 0.69314718
= log 2)

Objective function: 2
(gap = 3.6193271E-14 i.e.,
1.8096635E-12%)

Dual variables (of the dual GP problem) = y:

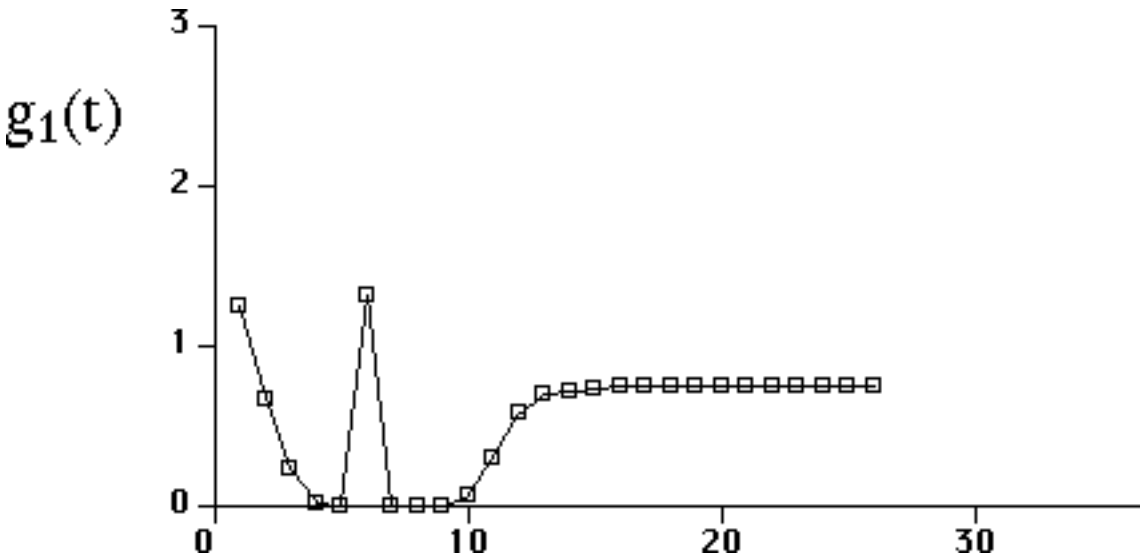
i	y[i]
1	-6.9314718E-1
2	1.5030399E0
3	-1.5030399E0
4	-1.0144972E-15

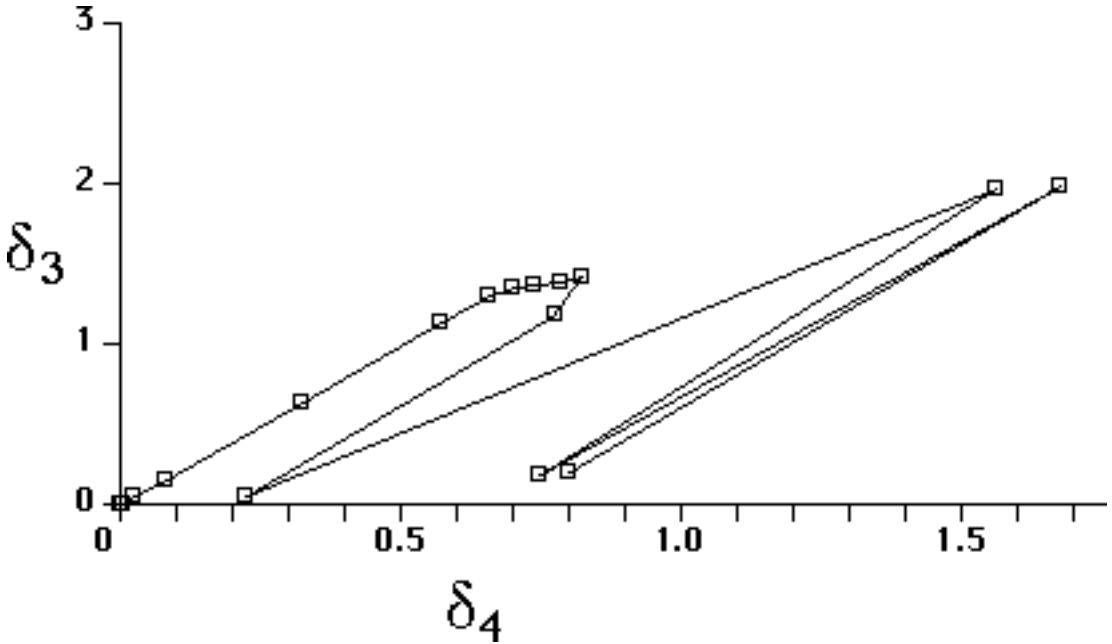
Primal GP solution (t):

i	t[i]
1	4.4953339E0
2	2.2245289E-1

Constraints:

Posy	Value	Infeasibility	Lambda
2	0.7525079477	0.0000000E0	6.3457218E-14


$g_1(t)$
3
2
1
0
0
10
20
30


3
2
1
0
δ_3
0
0.5
1.0
1.5
δ_4

The algorithm, because dual variables are kept positive, converges to a primal optimal solution!