Geometric Programming

Fanlin Wang

May 7, 2019

What is Geometric Programming?

- A geometric program (GP) is a type of optimization problem that has objective and constraint functions in a special form.
- Recall the convex optimization problem and linear programming (LP) we have learnt so far...

Convex optimization:

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq b_i, \\ & i = 1, \ldots, m \end{array}$$

Linear optimization:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & a_i^T(x) \leq b_i, \\ & i = 1, \ldots, m \end{array}$$

- Linear optimization problem is a special form of convex optimization. So is geometric program.

Monomial functions

Definition

Let $x_1, ..., x_n$ denote n real positive variables, and $x = (x_1, \ldots, x_n)$ a vector with components x_i. A monomial function is a real valued function f of x, with the form $f(x) = cx_1^{a_1}x_2^{a_2}\ldots x_n^{a_n}$, where $c > 0$ and $a_i \in R$. We call c the coefficient of the monomial and $a_1, \ldots, a_n$ the exponents of the monomial.

Check: are these functions monomial?

$$f(x) = 2$$

$$f(x) = 2.3x^2$$

Posynomial functions

Definition

A posynomial function is a sum of one or more monomials.

Check: are these functions posynomial?

$$f(x,y) = 0.23 + x/y$$
$$f(x,y) = 2(1+xy)^{3.1}$$
$$f(x,y) = x - y$$

Definition

A generalized posynomial function can be formed from posynomials using the operations of addition, multiplication, positive power, and maximum.

GP - Standard form

Definition

A geometric program (GP) is an optimization problem of the form

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 1, i = 1, \ldots, m \\ & g_i(x) = 1, i = 1, \ldots, p \end{array}$$

where f_i are posynomials and g_i are monomials. If f_i are generalized posynomials, then we call such problem a generalized geometric program (GGP).

- We have efficient and reliable solutions for practical large-scale GP problems.

GP - Extended forms

We can transform certain problems to an equivalent GP standard form.

Example

Original problem:

$$\begin{array}{ll} \text{maximize} & x/y \\ \text{subject to} & 2 \leq x \leq 3, x^2 + 3y/z \leq y^{1/2}, \\ & x/y = z^2 \end{array}$$

Transform it to the standard form:

$$\begin{array}{ll} \text{minimize} & x^{-1}y \\ \text{subject to} & 2x^{-1} \leq 1, (1/3)x \leq 1, \\ & x^2y^{-1/2} + 3y^{1/2}z^{-1} \leq 1, \\ & xy^{-1}z^{-2} = 1 \end{array}$$

A simple application

Example

We want to have a box with maximum volume, and its height h, width w, and depth d are subject to the following constraints:

1. limit on the floor area (wd): A_{flr}
2. limit on the wall area ($2(hw + hd)$): A_{wall}
3. lower and upper limits on the ratio h/w and w/d

Formulate the problem in terms of an optimization problem:

$$\begin{array}{ll} \text{maximize} & hwd \\ \text{subject to} & 2(hw + hd) \leq A_{wall}, wd \leq A_{flr} \\ & \alpha \leq h/w \leq \beta, \gamma \leq w/d \leq \delta \end{array}$$

A simple application

$$\begin{aligned} \text{maximize} \quad & hwd \\ \text{subject to} \quad & 2(hw + hd) \leq A_{wall}, wd \leq A_{flr}, \\ & \alpha \leq h/w \leq \beta, \gamma \leq w/d \leq \delta \end{aligned}$$

Verify that the problem can be converted to a GP standard form:

$$\begin{aligned} \text{minimize} \quad & h^{-1}w^{-1}d^{-1} \\ \text{subject to} \quad & 2(hw + hd)/A_{wall} \leq 1, wd/A_{flr} \leq 1, \\ & w\alpha/h \leq 1, h/(w\beta) \leq 1, \\ & d/(w\delta) \leq 1, w\gamma/d \leq 1 \end{aligned}$$

GGPLAB

1. a Matlab package for solving GPs
2. Installation:
   1. Download GGPLAB from:
      `https://web.stanford.edu/~boyd/ggplab/`
   2. Add the GGPLAB directory to Matlab's path:
      ```
      >> addpath /home/username/ggplab
      ```

GGPLAB - Basic Usage

1. GP variables

```
>> gpvar x y z w(25);
```

2. Monomials and posynomials

```
>> m1 = 2*x^3*y^4/z^5;
>> m1 = 1/2*sqrt(m1)*w(1)^2*w(2)^(-1)*w(20)^4.5;
>> p1 = 1 + x^1.5*y^(-4)*z^2 + m1 + 4*x*y;
>> p2 = p1/m2;
>> g1 = (4 + x^5.1 + m1)^2.4;
>> g2 = max( 1/x^2, m1^3, p1 );
```

3. Constraints

```
>> c1 = (x*y)^4 <= z^2;
>> c2 = p1 <= 1;
>> c3 = m1 == m2;
```

4. Solver

```
>> [obj_value, solution, status] =
gpsolve(obj, constr_array, type) % type can be 'min' or 'max'
```

Example

```
% problem constants
Awall  = 10000; Afloor = 1000;
alpha = 0.5; beta = 2; gamma = 0.5; delta = 2;

% GP variables
gpvar h w d

% objective function is the box volume
volume = h*w*d;

% set of constraints expressed as an array
constr = [ 2*(h*w + h*d) <= Awall; w*d <= Aflr;
           alpha <= h/w; h/w <= beta;
           gamma <= d/w; d/w <= delta;];

% solve the given GP problem
[max_volume solution status] = gpsolve(volume, constr, 'max');
assign(solution);
```

The result returned is:

```
>> Maximum volume is 77459.67.
>> Optimal solution is height h = 77.4597, width w = 38.7298, depth d = 25.8199.
```

Trade-off analysis

In the box example, consider the case when we want to change the wall area or the floor area, i.e., we want to investigate the maximum volume of the box under different constraints. Such kind of problem is called trade-off analysis. We change a GP in a standard form to a perturbed GP by replacing 1 on the right-hand side of each constraint with a parameter:

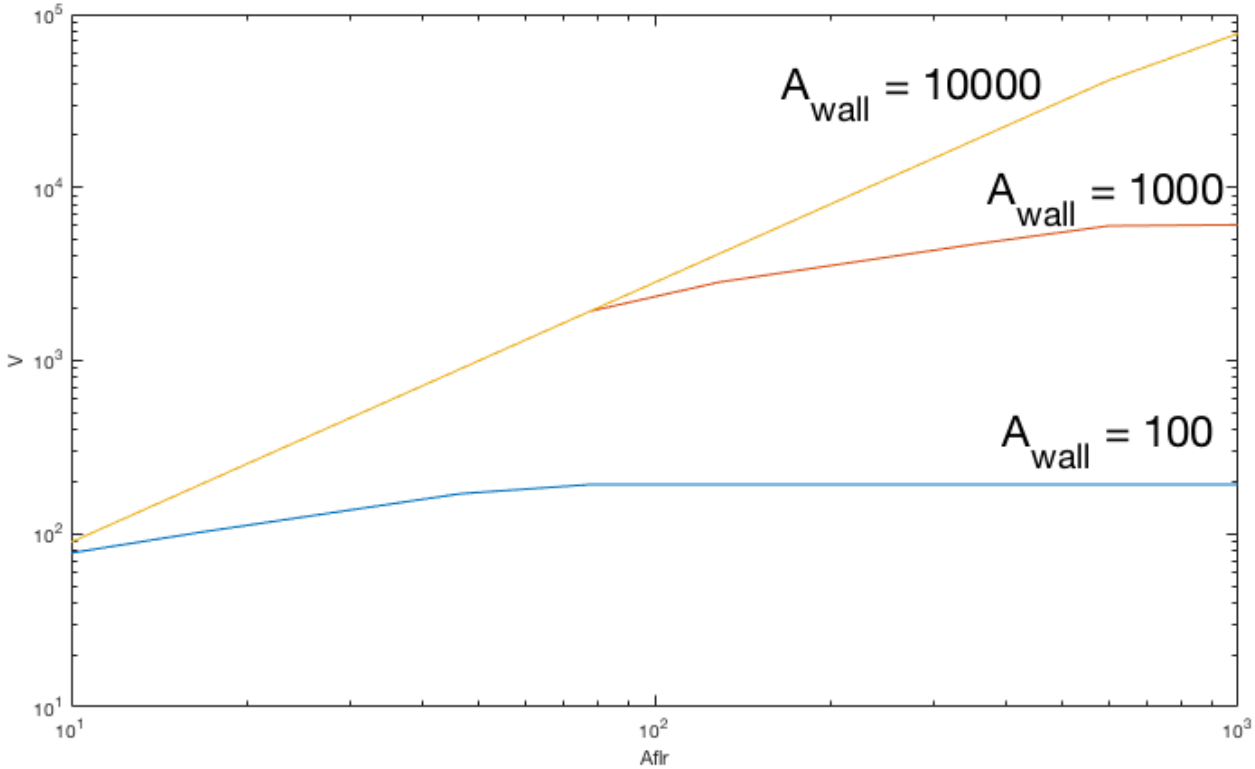
$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq u_i, i = 1, \ldots, m \\ & g_i(x) = v_i, i = 1, \ldots, p \end{array}$$

where u_i and v_i are positive constants. When $u_i > 1 (< 1)$, the ith inequality constraint is called *loosened*(*tightened*). Let $p(u, v)$ denote the optimal value of the perturbed GP, we can plot $p(u, v)$ against one or more constraints to get the trade-off curve.

Trade-off analysis on the box example

```
gpvar h w d

% varying parameters for an optimal trade-off curve
N = 10;
Aflr = logspace(1,3,N);
Awall = [100 1000 10000];
opt_volumes = zeros(length(Awall),N);

% setup various GP problems with varying parameters
for k = 1:length(Awall)
  for n = 1:N
    % change constraints with varying parameters
    constr(1) = 2*h*w + 2*h*d <= Awall(k);
    constr(2) = w*d <= Aflr(n);
    % solve the GP problem and compute the optimal volume
    [max_volume solution status] = gpsolve(volume, constr, 'max');
    opt_volumes(k,n) = max_volume;
  end
end

% plot the tradeoff curve
loglog(Aflr,opt_volumes(1,:), Aflr,opt_volumes(2,:), Aflr,opt_volumes(3,:));
xlabel('Aflr'); ylabel('V');
```

Trade-off analysis on the box example

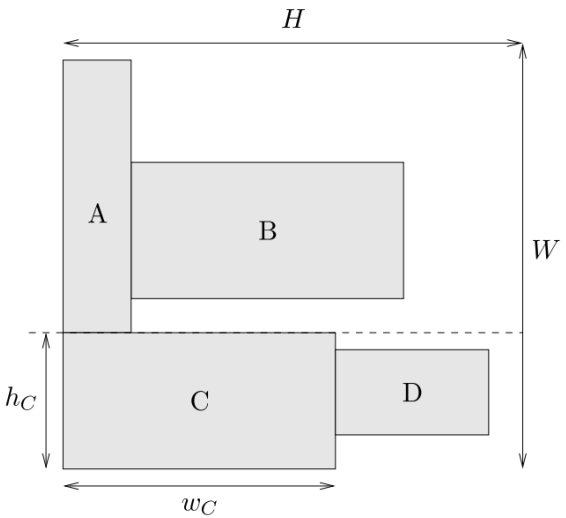


Example 2: floor planning

Example

We want to configure some rectangles such that they take up as little bounding space as possible. They are placed in a way that they do not overlap. Additional constraints can be:

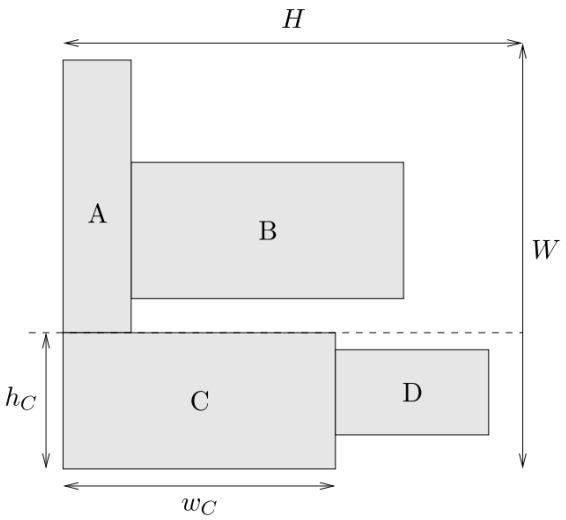
1. the area of each rectangle is fixed
2. the relative position of rectangles is fixed



Let's fix the relative position of four rectangles A, B, C, D:

1. A is to the left of B
2. C is to left of D
3. A, B are above C, D

Example 2: floor planning



Denote the width for A, B, C, D as w_A, w_B, w_C, w_D and the height as h_A, h_B, h_C, h_D, we have:

$$W = max\{w_A + w_B, w_C + w_D\}$$

$$H = max\{h_A, h_B\} + max\{h_C, h_D\}$$

The objective function is:

$$WH = max\{w_A + w_B, w_C + w_D\}(max\{h_A, h_B\} + max\{h_C, h_D\})$$

Example: floor planning

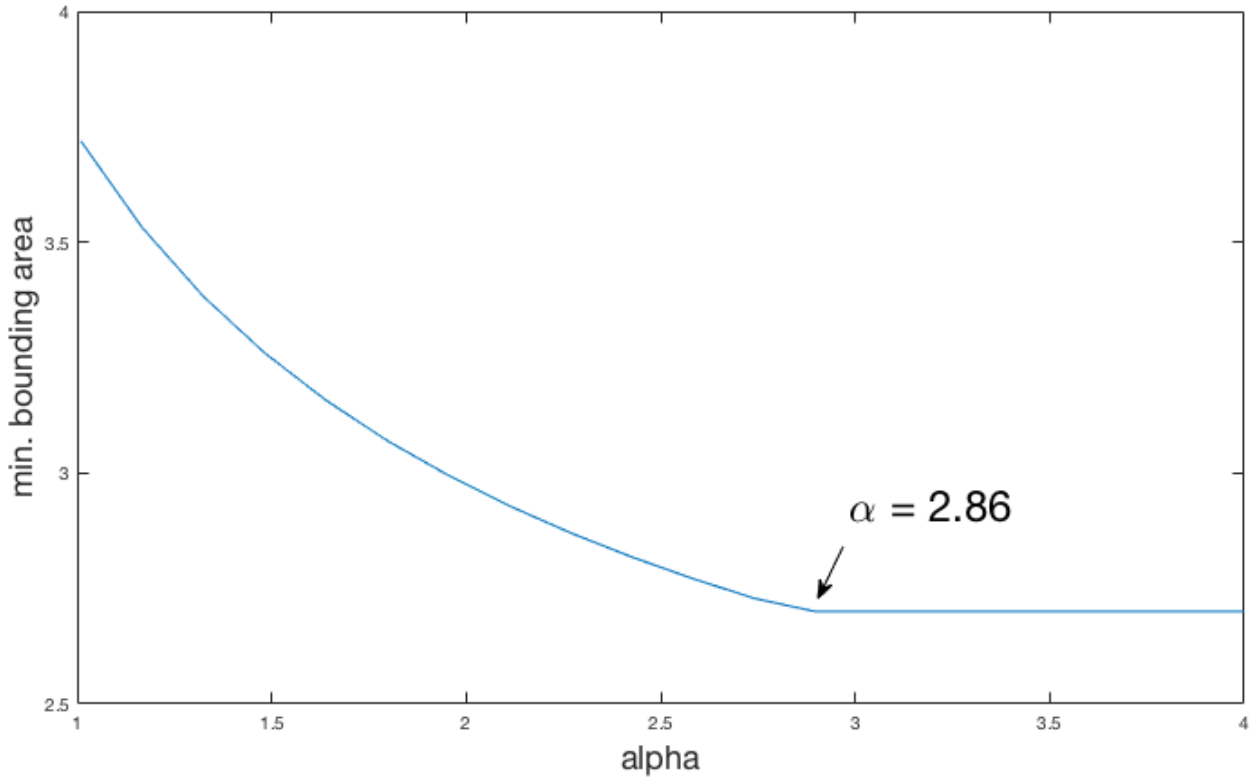
Formulate the problem in terms of an optimization problem:

$$\begin{aligned} \text{minimize} \quad & max\{w_A + w_B, w_C + w_D\}(max\{h_A, h_B\} + max\{h_C, h_D\}) \\ \text{subject to} \quad & h_A w_A = a, h_B w_B = b, h_C w_C = c, h_D w_D = d, \\ & \alpha \leq h/w \leq \beta, \gamma \leq w/d \leq \delta \end{aligned}$$

Solving GGP using GGPLAB: floor planning problem

```
% Let's assume the area for a,b,c,d are constants
a = 0.2; b = 0.5; c = 1.5; d = 0.5;

% GP variables
gpvar wa wb wc wd ha hb hc hd

% objective function is the area of the bounding box
obj = max(wa + wb, wc + wd)*(max(ha,hb) + max(hc,hd));

% constraints functions (for those constraint functions that have fixed parameter)
constr = [ ha*wa == a; hb*wb == b; hc*wc == c; hd*wd == d ];

% alpha is the changing parameter in the tradeoff analysis
N = 20;
alpha = linspace(1.01,4,N);

min_area = []; status_history = {};
for n = 1:N
  constr(5)  = 1/alpha(n) <= ha/wa; constr(6)  = ha/wa <= alpha(n);
  constr(7)  = 1/alpha(n) <= hb/wb; constr(8)  = hb/wb <= alpha(n);
  constr(9)  = 1/alpha(n) <= hc/wc; constr(10) = hc/wc <= alpha(n);
  constr(11) = 1/alpha(n) <= hd/wd; constr(12) = hd/wd <= alpha(n);
  [obj_value, solution, status] = gpsolve(obj, constr);
  min_area(n,1) = obj_value;
  status_history{end+1} = status;
end
```

Solving GGP using GGPLAB: floor planning problem



References

- Stephen Boyd, Seung-Jean Kim, Lieven Vandenberghe and Arash Hassibi. *A tutorial on geometric programming* https://web.stanford.edu/~boyd/papers/pdf/gp_tutorial.pdf, date = 2006,
- Stephen Boyd and Lieven Vandenberghe. *Convex Optimization* http://web.stanford.edu/~boyd/cvxbook/, date = 2006,
- Almir Mutapcic. *Floor planning example* https://web.stanford.edu/~boyd/ggplab/examples/floor_planning.m, date = 2006,
- Almir Mutapcic. *Box volume maximization example* https://web.stanford.edu/~boyd/ggplab/examples/max_volume_box.m, date = 2006,