KOMPUTASI KAPASITAS SALURAN *(CHANNEL CAPACITY)* DENGAN PEMROGRAMAN GEOMETRIK DALAM BENTUK KONVEKS

SKRIPSI

Diajukan Guna Melengkapi Tugas Akhir dan Memenuhi Syarat-syarat
Untuk Menyelesaikan Program Studi Matematika (S1)
dan Mencapai Gelar Sarjana Sains

Oleh :

TITIN SYAHADATINA
NIM : 031810101028

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2007

PERSEMBAHAN

Skripsi ini saya persembahkan untuk :

1. Almamater Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember;
2. Ibuku Siti Maimunah dan Bapakku Samikudin tercinta, terimakasih atas dorongan semangat, doa, kasih sayang, dan pengorbanan yang takkan pernah terbalas oleh apapun didunia ini ;
3. Nenekku Suciati tercinta, kakak-kakakku Sugiono, Sugianto(alm), Sulasmono, Rahmat Santoso, Moh. Hafidudin serta adik-adikku kalian adalah pemacu semangatku;
4. Guru-guruku yang telah memberikan ilmu dan membimbing dengan penuh kesabaran.

MOTTO

...Sesungguhnya Allah tidak merubah keadaan sesuatu kaum sehingga mereka merubah keadaan yang ada pada diri mereka sendiri...

(Terjemahan surat AR-Ra'd: 11)

...Sesungguhnya sesudah kesulitan itu ada kemudahan...

(Terjemahan Alam Nasyrah:6)

Kesuksesan adalah milik seseorang yang berlari bukan orang yang duduk diam apalagi tidur

(Harfa)

Ikhtiar dan tawakkal adalah dua sisi yang tak dapat dipisahkan dalam hidup ini
Jangan pernah berhenti berusaha sampai waktu benar-benar telah habis
Jangan pernah putus asa dari rahmat Allah SWT., karena
Kekuatan hanya milik Allah SWT.

(TiSya)

PERNYATAAN

Saya yang bertanda tangan dibawah ini:

Nama : Titin Syahadatina

NIM : 031810101028

Menyatakan dengan sesungguhnya bahwa karya tulis ilmiah yang berjudul: "Komputasi Kapasitas Saluran *(Channel Capacity)* Dengan Pemrogramam Geometrik Dalam Bentuk Konveks" adalah benar-benar karya sendiri, kecuali jika disebutkan sumbernya dan belum pernah diajukan pada institusi manapun, serta bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa adanya paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata dikemudian hari pernyataan ini tidak benar.

Jember, Juni 2007

Yang menyatakan,

Titin Syahadatina

031810101028

SKRIPSI

KOMPUTASI KAPASITAS SALURAN *(CHANNEL CAPACITY* DENGAN PEMROGRAMAN GEOMETRIK DALAM BENTUK KONVEKS

Oleh

Titin Syahadatina

NIM 031810101028

Pembimbing

Dosen Pembimbing Utama : Agustina Pradjaningsih, S.Si., M.Si.

Dosen Pembimbing Anggota : Ahmad Kamsyakawuni, S.Si.

PENGESAHAN

Skripsi berjudul *Komputasi Kapasitas Saluran (Channel Capacity) Dengan Pemrogramam Geometrik Dalam Bentuk Konveks* telah diuji dan disahkan oleh Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember pada:

Hari : SENIN

Tanggal : 09 JUL 2007

Tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Tim Penguji:

Ketua,

Agustina Pradjaningsih, S.Si., M.Si.
NIP 132 257 933

Sekretaris,

Ahmad Kamsyakawuni, S.Si.
NIP. 132 206 038

Anggota I,

Kusbudiono, S.Si
NIP. 132 314 577

Anggota II,

Kristiana Wijaya, S.Si. M.Si
NIP. 132 258 180

Mengesahkan
Dekan FMIPA Universitas Jember

Ir. Sumadi, MS.
NIP. 130 368 784

RINGKASAN

Komputasi Kapasitas Saluran (*Channel Capacity*) Dengan Pemrograman Geometrik Dalam Bentuk Konveks, Titin Syahadatina, 031810101028, 2007,. Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Kapasitas saluran *(channel capacity)* merupakan dasar dari teori informasi yang ditemukan oleh Shanon. Kapasitas saluran dapat digunakan untuk menentukan jumlah informasi maksimum per detik yang dikirimkan melalui saluran komunikasi. Penentuan kapasitas saluran merupakan salah satu acuan dalam masalah pengiriman data yang dikirim sehingga data yang dikirimkan tidak melebihi kapasitasnya yang berpengaruh pada penerimaan pesan yang akan diterima oleh penerima.

Kapasitas saluran dapat diselesaikan dengan memodelkannya dalam bentuk pemrograman geometrik dalam bentuk konveks dan dari model yang telah ditentukan, kapasitas saluran dikomputasi. Masalah kapasitas saluran diskret tanpa memori yang akan dikomputasi ada dua yaitu kapasitas saluran tanpa input cost dan kapasitas saluran dengan input cost. Komputasi pada masalah kapasitas saluran ini dilakukan dengan menggunakan *ggplab* dan *gpcvx* pada MATLAB.

Hasil komputasi dengan MATLAB adalah nilai laju informasi maksimum yang digunakan untuk menentukan nilai maksimum kapasitas saluran baik pada kapasitas saluran tanpa input cost maupun pada kapasitas saluran dengan input cost. Laju informasi maksimum ini dipengaruhi oleh nilai dan ukuran matriks probabilitas **P** yang menyatakan nilai probabilitas pesan yang dikirimkan dan ukuran dari pesan yang dikirimkan. Hubungan yang diperoleh antara matriks transisi probabilitas dan nilai laju informasi maksimum yang dapat dikirimkan melalui saluran komunikasi berbanding terbalik dengan matriks transisi probabilitas **P**.

PRAKATA

Puji syukur Alhamdulillah penulis panjatkan kepada Allah SWT. yang selalu melimpahkan rahmat dan karunia-NYA, sehingga penulis dapat menyelesaikan karya tulis ilmiah yang berjudul " Komputasi Kapasitas Saluran *(Channel Capacity)* Dengan Pemrograman Geometrik Dalam Bentuk Konveks". karya tulis ilmiah ini disusun untuk memenuhi salah satu syarat dalam menyelesaikan pendidikan strata satu (S1) pada Jurusan Matematika, Fakultas MIPA, Universitas Jember.

Penyusunan karya tulis ilmiah ini tidak lepas dari bantuan berbagai pihak, oleh karena itu penulis ingin menyampaikan ucapan terima kasih kepada :

1. Ibu Agustina Pradjaningsih, S.Si., M.Si., selaku Dosen Pembimbing Utama dan Bapak Ahmad Kamsyakawuni, S.Si., selaku Dosen Pembimbing Anggota yang telah meluangkan waktu dan pikiran serta perhatiannya guna memberikan bimbingan dan pengarahan demi terselesaikannya penulisan karya tulis ilmiah ini;
2. Bapak Kusbudiono S.Si., selaku Dosen Penguji I dan Ibu Kristiana Wijaya.S.Si. M.Si.,selaku Dosen Penguji II yang telah memberikan arahan dan kritikan demi kesempurnaan penulisan karya tulis ilmiah ini;
3. Bapak dan ibuku, serta keluargaku yang telah memberikan dorongan semangat dan doa demi terselesainya karya tulis ilmiah ini;
4. Saudara-saudaraku "IONS" dan "Jalak1" serta teman-teman seangkatan seperjuangan "Matematika 2003" terimakasih untuk kalian semua.

Pada akhirnya penulis mohon maaf apabila dalam penulisan dan penyusunan karya tulis ilmiah ini terdapat kelalaian. Semoga karya ini bermanfaat.

Jember, Juni 2007

Penulis

DAFTAR ISI

HALAMAN JUDUL i
HALAMAN PERSEMBAHAN ii
HALAMAN MOTTO iii
HALAMAN PERNYATAAN iv
HALAMAN PEMBIMBINGAN v
HALAMAN PENGESAHAN vi
RINGKASAN vii
PRAKATA viii
DAFTAR ISI ix
DAFTAR GAMBAR xi
BAB.1 PENDAHULUAN
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Tujuan 2
1.4 Manfaat 2
BAB 2. TINJAUAN PUSTAKA
2.1 Fungsi Monomial Dan Fungsi Posinomial 3
2.1.1 Bentuk Umum Fungsi Monomial 3
2.1.2 Bentuk Umum Fungsi Posinomial 3
2.2 Pemrograman Geometrik 4
2.2.1 Pemrograman Geometrik Dalam Bentuk Standar 4
2.2.2 Pemrograman Geometrik Dalam Bentuk Konveks 4
2.3 Himpunan dan Fungsi Konveks 5
2.4 Kapasitas Saluran (*Channel Capacity*) 7

2.5 Bentuk Pemrograman Geometrik Masalah Kapasitas Saluran 9
2.5.1 Kapasitas Saluran dengan Input Cost 9
2.5.2 Kapasitas Saluran Tanpa Input Cost 10
2.6 Langkah-Langkah Penyelesaian 10
BAB 3. HASIL DAN PEMBAHASAN
3.1 Nilai Input Awal 11
3.2 Kapasitas Saluran Tanpa Input Biaya 12
3.3 Kapasitas Saluran Dengan Input Biaya 22
BAB 4. KESIMPULAN DAN SARAN
4.1 Kesimpulan 31
4.2 Saran 31
DAFTAR PUSTAKA 32
LAMPIRAN 33

DAFTAR GAMBAR

2.1 Himpunan konveks dan non konveks 5
2.2 Fungsi konveks dan fungsi konkaf 6
3.1 Gap Untuk Probabilitas = [0,3 0,2 0,5] Tanpa Input Cost 18
3.2 Gap Untuk Probabilitas = [0,25 0,25 0,2 0,15 0,15] Tanpa Input Cost 21
3.3 Gap Untuk Probabilitas = [0,3 0,2 0,5] Dengan Input Cost 27
3.4 Gap untuk Probabilitas = [0,25 0,25 0,2 0,15 0,15] Dengan Input Cost 29

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Pesatnya perkembangan teknologi komunikasi menyebabkan banyaknya penelitian mengenai sistem komunikasi. Salah satu hal yang sangat berhubungan dengan sistem komunikasi adalah transmisi informasi. Transmisi informasi dikirim melalui saluran komunikasi yang mempunyai kapasitas yang berbeda untuk mengirimkan informasi dari sumber ke penerima pesan.

Kapasitas dari saluran komunikasi disebut kapasitas saluran *(channel capacity)*. Kapasitas saluran merupakan dasar dari teori informasi yang ditemukan oleh Shannon. Kapasitas saluran dapat menentukan jumlah informasi maksimum per detik yang dikirimkan melalui saluran komunikasi dengan satuan bit per detik yang dapat diperoleh dengan memaksimumkan rata-rata informasi timbal balik *(mutually information)* (Blahut,1972).

Kapasitas saluran *(channel capacity)* merupakan salah satu masalah sistem komunikasi yang dapat diselesaikan dengan memodelkannya dalam bentuk pemrograman geometrik yang merupakan salah satu kelas dari masalah pemrograman nonlinier. Untuk mendapatkan nilai yang optimum pemrograman geometrik dari kapasitas saluran diubah dalam bentuk konveks (Wahyuni, 2007).

Dalam teori informasi jumlah informasi yang dikirimkan melalui saluran komunikasi jumlahnya harus lebih kecil dari nilai kapasitas salurannya (Papoulis, 1992). Untuk mendapatkan nilai optimum dari kapasitas saluran tidak cukup hanya dengan memodelkan kapasitas saluran dalam bentuk pemrograman geometrik, tetapi juga diperlukan sebuah komputasi dari model kapasitas saluran dengan pemrograman geometrik dalam bentuk konveks.

Pada banyak kasus penyelesaian masalah kapasitas saluran secara analitik cukup sulit, membutuhkan waktu dan pengerjaan yang panjang untuk menemukan solusinya dalam skala kecil bahkan tidak dapat ditemukan solusinya dalam skala yang besar. Oleh karena itu skripsi ini memformulasikan kembali masalah komputasi kapasitas saluran dengan pemrograman geometrik dalam bentuk konveks.

1.2 Rumusan Masalah

Berdasarkan uraian pada latar belakang penulis dapat menentukan rumusan masalah dari skripsi ini yaitu bagaimana mengkomputasi kapasitas saluran dengan pemrograman geometrik dalam bentuk konveks pada saluran diskret tanpa memori menggunakan program komputer.

1.3 Tujuan

Tujuan dari skripsi ini adalah untuk menganalisa laju informasi maksimum yang dapat dikirimkan melalui saluran diskret tanpa memori yang digunakan untuk menentukan kapasitas maksimum saluran melalui aplikasi program komputer.

1.4 Manfaat

Manfaat dari skripsi ini adalah memberikan pengetahuan kepada mahasiswa mengenai kapasitas saluran komunikasi dan memberikan informasi kepada pihak-pihak yang berhubungan dengan sistem komunikasi mengenai cara untuk mendapatkan nilai laju informasi maksimum yang bisa digunakan sebagai salah satu acuan dalam masalah pengiriman data agar data yang dikirimkan benar sesuai dengan kapasitasnya.

BAB 2. TINJAUAN PUSTAKA

Optimasi kapasitas saluran digunakan untuk mendapatkan nilai optimum kapasitas saluran. Kapasitas saluran yang akan digunakan dalam optimasi adalah kapasitas saluran dengan pemrograman geometrik dalam bentuk konveks. Berikut beberapa dasar dalam masalah pemrograman geometrik dari kapasitas saluran dalam bentuk konveks:

2.1 Fungsi Monomial dan Posinomial

2.1.1 Bentuk Umum Fungsi Monomial

Fungsi monomial $f(\mathbf{x})$ didefinisikan sebagai berikut :

$$f(\mathbf{x}) = x_1^{a_1} x_2^{a_2} \ldots x_n^{a_n} \tag{2.1}$$

dengan fungsi $f : \mathbf{R}^n \rightarrow \mathbf{R}$, domain $f : \mathbf{R}_+^n$

dimana: c = konstanta pengali, $c > 0$

a_j = konstanta eksponensial dan $a_j \in \mathbf{R}$, $j = 1,2,\ldots n$

(Boyd & Vandenberghe, 2004).

2.1.2 Bentuk Umum Fungsi Posinomial

Fungsi posinomial $f(\mathbf{x})$ didefinisikan sebagai penjumlahan dari fungsi –fungsi monomial yang ditulis dalam persamaan berikut:

$$f(\mathbf{x}) = \sum_{k=1}^{K} c_k x_1^{a_{1k}} x_2^{a_{2k}} \ldots x_n^{a_{nk}} \tag{2.2}$$

dengan $x \succ 0$ dan $c_k > 0$, $a_{jk} \in \mathbf{R}$, $k = 1,2,\ldots, K$, j=1,2,…,n

(Boyd & Vandenberghe, 2004).

2.2 Pemrograman Geometrik

2.2.1 Pemrograman Geometrik dalam Bentuk Standar

Pemrograman geometrik merupakan salah satu kelas dari pemrograman nonlinier. Pemrograman geometrik digunakan untuk meminimumkan fungsi tujuan yang berbentuk posinomial ke batas atas pertidaksamaan posinomial dari fungsi kendala dan persamaan fungsi kendala yang berbentuk monomial.

minimumkan $f_0(\mathbf{x})$

dengan kendala $f_i(\mathbf{x}) \le 1,\ i = 1,\ldots,N$

$h_j(\mathbf{x}) = 1,\ j = 1,\ldots,M$ (2.3)

dengan f_i adalah posinomial dan h_j adalah monomial.

Contoh masalah pemrograman geometrik dalam bentuk standar :

minimumkan $f(x_1,x_2,x_3) = x_1(x_2 + x_3)$

kendala $x_1^2\sqrt{x_2x_3} \ge 0.8, \sqrt{x_1x_2} \ge 0.5, x_1 \ge 1$

(Chiang & Boyd, 2003).

2.2.2 Pemrograman Geometrik Dalam Bentuk Konveks

Pemrograman geometrik dalam bentuk konveks diperoleh dari hasil transformasi fungsi monomial dan fungsi posinomial yaitu sebagai berikut :

Pada transformasi variabel yang akan digunakan adalah $y_i = log\ x_i$ sehingga $x_i = e^{yi}$.

Jika f adalah monomial maka hasil transformasi f adalah:

$$f(x) = f(e^{y1}, \ldots, e^{yn}) = e^{a^T y + b},\ b = log\ c \quad (2.4)$$

Jika f adalah posinomial maka transformasi f adalah:

$$f(x) = \sum_{k=1}^{K} e^{a_k^T y + b_k} \quad (2.5)$$

Sehingga dengan transformasi logaritma dari persamaan (2.3), (2.4) dan (2.5) diperoleh bentuk pemrograman geometrik dalam bentuk konveks yaitu:

Minimumkan $\log \sum_{k=1}^{K_0} e^{a_{0k}^T y + b_{0k}}$ (2.6)

Kendala $\log \sum_{k=1}^{K_i} e^{a_{ik}^T y + b_{ik}} \leq 0, i = 1,...,m$

$g_i^T y + h_i = 0, g_i \in \mathrm{R}^n, i = 1,...,M$

(Boyd & Vandenberghe, 2004).

2.3 Himpunan dan Fungsi Konveks

Beberapa dasar yang digunakan dalam pemrograman konveks adalah himpunan konveks dan fungsi konveks.

2.3.1 Himpunan Konveks

Sebuah himpunan titik di S adalah konveks jika segmen garis di antara dua titik di S dengan $S \subseteq \mathbf{R}^n$ berada didalam S. Secara matematis dapat dituliskan sebagai berikut:

Jika titik $\mathbf{x_1}, \mathbf{x_2} \in S$ dengan $S \subseteq \mathbf{R}^n$, terdapat α sehingga titik $\mathbf{x}$ dapat dituliskan sebagai

$$\mathbf{x} = \alpha \mathbf{x_1} + (1-\alpha)\mathbf{x_2}, \quad 0 \leq \alpha \leq 1 \tag{2.6}$$

(Boyd & Vandenberghe, 2004).

Teorema 2.1

Interseksi dari beberapa himpunan konveks juga himpunan konveks. Berikut ini contoh himpunan konveks dan himpunan non konveks:

a. Himpunan konveks b. Himpunan non konveks

Gambar 2.1 Himpunan konveks dan non konveks

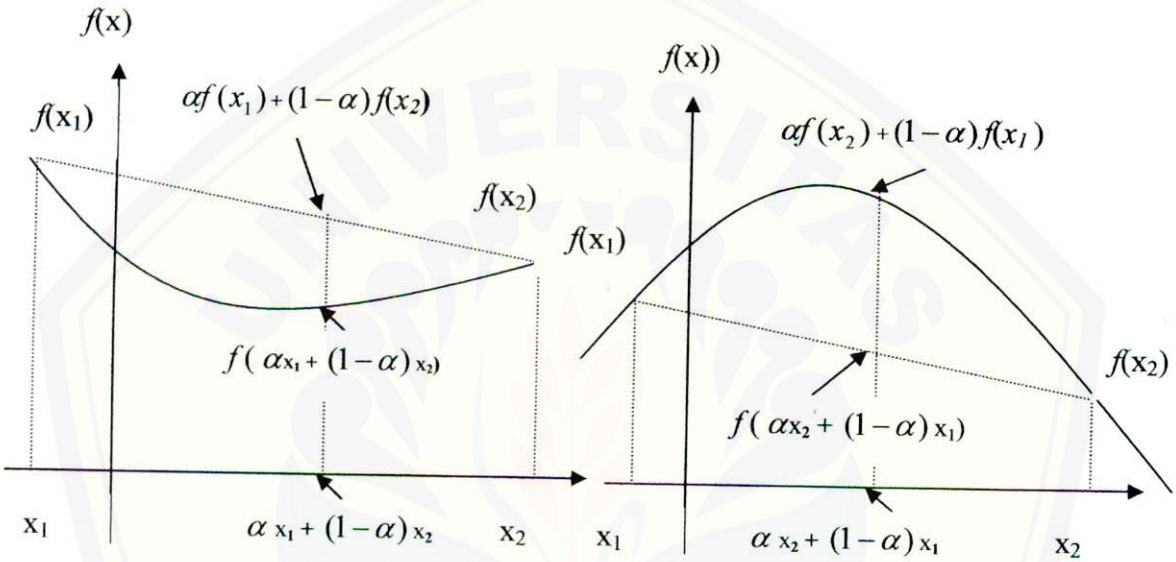
Pada Gambar 2.1 dapat dilihat segmen garis dari dua titik pada himpunan konveks ditunjukkan sebagai titik-titik di dalam himpunan. Sedangkan pada himpunan non konveks terdapat segmen garis dari dua titik dengan titik-titik tertentu berada di luar himpunan.

2.3.2 Fungsi Konveks

Salah satu ciri sebuah fungsi f dikatakan konveks jika domain fungsi f adalah himpunan konveks. Secara matematis fungsi konveks didefinisikan sebagai berikut. Jika untuk setiap dua titik yang berbeda $\mathbf{x_1}, \mathbf{x_2} \in S$ dan untuk semua $0 \leq \alpha \leq 1$, maka

$$f(\alpha \mathbf{x_1} + (1-\alpha)\mathbf{x_2}) \leq \alpha f(\mathbf{x_1}) + (1-\alpha) f(\mathbf{x_2}) \qquad (2.7)$$

Contoh fungsi konveks dan fungsi konkaf terdapat pada Gambar 2.2



a. Fungsi konveks

b. Fungsi konkaf

Gambar 2.2 Fungsi konveks dan fungsi konkaf

Pada Gambar 2.2 dapat dilihat bahwa fungsi konkaf adalah fungsi negatif dari fungsi konveks.

Teorema 2.2

Misalkan S himpunan konveks tak kosong dalam R^n, dan misalkan $f: S \rightarrow R^1$ differensiabel pada S. Maka f adalah konveks jika $\forall x_1, x_2 \in S$ berlaku

$$f(\mathbf{x_2}) \geq \mathbf{x_1} + \nabla f^T(\mathbf{x_1})(\mathbf{x_2} - \mathbf{x_1}) \qquad (2.8)$$

dengan $\nabla f(\mathbf{x_1})$ adalah gradien fungsi f.

Teorema 2.3

Penjumlahan dari fungsi-fungsi konveks adalah fungsi konveks juga. Secara matematis, jika $f_j(\mathrm{x})$ adalah fungsi-fungsi konveks pada himpunan S, maka fungsi

$$f(\mathbf{x}) = \sum_{j=1}^{k} f_j(\mathbf{x}), \quad j = 1,2,...,k \text{ juga konveks pada } S. \tag{2.9}$$

Teorema 2.4

Suatu minimum lokal dari suatu fungsi konveks $f(\mathrm{x})$ adalah minimum global. Juga dapat dibuktikan bahwa maksimum lokal dari suatu fungsi konkaf $f(\mathrm{x})$ adalah maksimum global.

(Teorema dan bukti dapat dilihat pada: Fitriya, 2003)

2.4 Kapasitas Saluran (*Channel Capacity*)

Kapasitas suatu saluran komunikasi didefinisikan sebagai jumlah maksimum informasi per detik (bit per detik) yang dapat ditransmisi oleh suatu sistem (saluran) (Misca, 1986).

Definisi 2.1

Saluran diskret merupakan sebuah sistem yang terdiri dari input X, output Y dan matriks transisi peluang $P(Y|X)$ yang menyatakan peluang pengamatan pada output diterima Y ketika pada input dikirim X. Sebuah saluran disebut tanpa memori jika distribusi peluang output tergantung hanya pada input pada saat itu dan dengan kondisi tidak terikat pada output atau input saluran sebelumnya.

Definisi 2.2

Kapasitas saluran diskret tanpa memori dinyatakan sebagai

$$C = \max_{p(x)} I(X;Y) \tag{2.10}$$

dimana yang dimaksimumkan adalah semua distribusi peluang input $p(x)$, dengan C adalah kapasitas saluran dan $I(X;Y)$ adalah informasi timbal balik antara input X dan output Y.

Dalam teori informasi kapasitas $C(S)$ dari saluran diskret tanpa memori dengan batasan nilai input $\mathbf{E_p[s]} = \mathbf{ps} \leq S$ adalah :

$$C(S) = \max_{\mathbf{p}:\mathbf{ps}\leq S} I(X;Y) \tag{2.11}$$

dengan $I(X;Y)$ adalah informasi timbal balik antar input X dan output Y.

Definisi 2.2

Informasi timbal balik antar input X dan output Y didefinisikan sebagai:

$$I(X;Y) = \sum_{i=1}^{N}\sum_{j=1}^{M} Q_{ij} \log \frac{Q_{ij}}{p_i q_j} = H(Y) - H(Y|X) = -\sum_{j=1}^{M} q_j \log q_j - \mathbf{pr} \tag{2.12}$$

dimana $\mathbf{r} \in \mathbf{R}^{N\times 1}$ dan $r_i = -\sum_{j=1}^{M} P_{ij} \log P_{ij}$ adalah entri dari r yang merupakan kondisi entropi dari Y jika diberikan $X = i$.

Dari persamaan (2.11) dan (2.12), kapasitas saluran sebagai nilai tujuan optimal dari masalah maksimisasi disebut sebagai masalah kapasitas saluran dengan nilai input yaitu :

maksimumkan $\quad -\mathbf{pr} - \sum_{j=1}^{M} q_j \log q_j$

dengan kendala $\quad \mathbf{pP} = \mathbf{q}, \quad \mathbf{ps} \leq S$ (2.13)

$\mathbf{p1} = 1, \quad \mathbf{p} \succeq \mathbf{0}$

dengan: $\mathbf{p}$ dan $\mathbf{q}$ adalah variabel optimasi,

$\mathbf{P}$ adalah matriks saluran yang merupakan parameter konstan,

$\mathbf{p} \succeq \mathbf{0}$ artinya $p_i \geq 0$, i=1,2,…,N.

Pada kasus khusus tidak ada batasan nilai input, masalah kapasitas saluran menjadi:

maksimumkan $\quad -\mathbf{pr} - \sum_{j=1}^{M} q_j \log q_j$

dengan kendala $\quad \mathbf{pP} = \mathbf{q},$ (2.14)

$\mathbf{p1} = 1, \quad \mathbf{p} \succeq \mathbf{0}$

(Wahyuni, 2007).

2.5 Bentuk Pemrograman Geometrik Kapasitas Saluran

Pemrograman geometrik kapasitas saluran diperoleh dari dualitas Lagrange kapasitas saluran. Berikut bentuk dualitas Lagrange dari kapasitas saluran:

2.5.1 Kapasitas Saluran dengan Input Cost

minimumkan $\log \sum e^{\alpha_j + \gamma S}$ (2.15)

dengan kendala $\mathbf{P}\alpha + \gamma \mathbf{s} \succeq -\mathbf{r}, \ \gamma \geq 0$

dengan α dan γ adalah variabel optimasi, $\mathbf{s}$ adalah parameter konstan yang berupa vektor kolom, S adalah parameter konstan dan $\mathbf{P}$ adalah matriks saluran yang merupakan parameter konstan, $\mathbf{r} \in \mathrm{R}^{N \times 1}$.

Pemrograman geometrik pada persamaan (2.15) dapat diubah ke dalam bentuk standar, melalui perubahan sifat eksponen pada variabelnya yaitu $z_j = \mathrm{e}^{\alpha_j}$ dan fungsi sasaran dualnya adalah:

minimumkan $w^S \sum_j z_j$

dengan kendala $w^{si} \prod_{j=1}^{M} z_j^{P_{ij}} \geq e^{-H(P^{(i)})}, i = 1,2,...,N$ (2.16)

$w \geq 1, \ z_j \geq 0, \ j = 1,2,...,M, \ w \geq 1$

dengan z, w adalah variabel optimasi dan $\mathbf{P}^{(i)}$ adalah baris ke i dari $\mathbf{P}$.

2.5.2 Kapasitas Saluran Tanpa Input Cost

Dual Lagrange dari kapasitas saluran tanpa nilai input dalam pemrograman geometrik bentuk konveks dapat dituliskan sebagai:

minimumkan $\log \sum_{j=1}^{M} e^{\alpha_j}$ (2.17)

dengan kendala $\mathbf{P}\alpha \succeq -\mathbf{r}$

dengan α adalah variabel optimasi dan $\mathbf{P}$ adalah matriks saluran yang merupakan parameter konstan.

Permasalahan dual Lagrange dari kapasitas saluran tanpa nilai input memiliki bentuk yang ekivalen dalam pemrograman geometrik bentuk standar yaitu:

minimumkan $\sum_{j=1}^{M} z_j$

dengan kendala $\prod_{j=1}^{M} z_j^{P_{ij}} \geq e^{-H(\mathbf{P}^{(i)})},\ i = 1,2,...,N,$ (2.18)

$z_j \geq 0,\ j = 1,2,...,M$

dengan z adalah variabel optimasi, $\mathbf{P}$ adalah matriks saluran yang merupakan parameter konstan, dan $\mathbf{P}^{(i)}$ adalah baris ke i dari $\mathbf{P}$ (Ciang, 2003).

2.6 Langkah-Langkah Penyelesaian

Langkah-langkah dalam menyelesaikan masalah optimasi kapasitas saluran dengan pemrograman geometrik dalam bentuk konveks adalah sebagai berikut :

1. Menentukan fungsi objektif dan kendala kapasitas saluran dalam bentuk pemrograman geometrik standar.
2. Menentukan matriks **A** terdiri dari $\mathbf{A}^0,\mathbf{A}^1,\ldots,\mathbf{A}^n$. Matriks **A** merupakan matriks yang disusun dari pangkat variabel optimum pada setiap monomial pada fungsi objektif dan kendala.
3. Menentukan matriks **b** terdiri dari $\mathbf{b}^0,\mathbf{b}^1,\ldots,\mathbf{b}^n$. Matriks **b** merupakan matriks yang disusun dari logaritma koefisien setiap monomial pada fungsi objektif dan kendala.
4. Menentukan ukuran dimensi dari **A**.
5. Fungsi standar kapasitas saluran menjadi bentuk konveks.
6. Komputasi variabel y sebagai variabel optimum yang baru.
7. Menentukan nilai optimum fungsi objektif.

BAB.4 KESIMPULAN DAN SARAN

4.1 Kesimpulan

Berdasarkan hasil yang diperoleh dapat disimpulkan nilai laju informasi maksimum yang dapat dikirimkan melalui saluran digunakan untuk menentukan nilai maksimum kapasitas saluran. Nilai laju informasi maksimum ini dipengaruhi oleh nilai dan ukuran matriks transisi probabilitas **P**. Semakin besar ukuran matriks transisi **P** maka nilai laju informasi maksimum yang dapat dikirimkan melalui saluran atau nilai maksimum kapasitas saluran akan mendekati 0. Sebaliknya semakin kecil matriks transisi **P** maka nilai laju informasi maksimum yang dapat dikirimkan melalui saluran atau nilai maksimum kapasitas saluran semakin besar.

4.2 Saran

Komputasi kapasitas saluran pada skripsi ini hanya terbatas pada saluran diskret tanpa memori dengan MATLAB, sehingga penelitian ini masih dapat dikembangkan misalnya pada saluran kontinu atau campuran dengan software selain MATLAB.

DAFTAR PUSTAKA

Boyd, S & Vandenberg, L. 2004. *Convex Optimization.*Cambride University.

Bazaraa, M., S., & Shetty, C.M. 1996. *Non Linier Programming: Theory and Algorithms*. New York: Willey.

Chiang, M & Boyd, S. 2003. *Geometric Programming Duals of Channel Capacity and Rate Distortion.* online.http://www.Stanford.edu/~boyd/reports/gp_chan_cap.pdf

Fitriya. 2003. *Studi Masalah Pemrograman Geometric dengan Metode Dual*. Jember: Laboratorium Komputer Jurusan Matematika Fakultas Matematika Dan Ilmu Pengetahuan Alam Universitas Jember.

M., Almir, K., Kwangmoo, S., Kim, Boyd, S. 2006. *GGPLAB: A Simple Matlab Toolbox For Geometric Programming Version 1.00.* online. http://www.Stanford.edu/~boyd/ggplab.

Papoulis, Athanasios.1992. *Probabilitas, Variabel Random, Dan Proses Stokastik.* Yogyakarta: Gajah Mada University Press.

Schwartz, Misca.1986. *Transmisi Informasi, Modulasi, dan Bising Terjemehan Sri Jatno Wirjosoedirjo*. Jakarta: Erlangga.

Wahyuni, Sri Endang. 2007. *Studi Masalah Pemrograman Geometrik Pada Kapasitas Saluran (Channel Capacity) dengan Dualitas Lagrange*. Jember: Laboratorium Komputer Jurusan Matematika Fakultas Matematika Dan Ilmu Pengetahuan Alam Universitas Jember.

LAMPIRAN A

Program kapasitas saluran tanpa inputcost

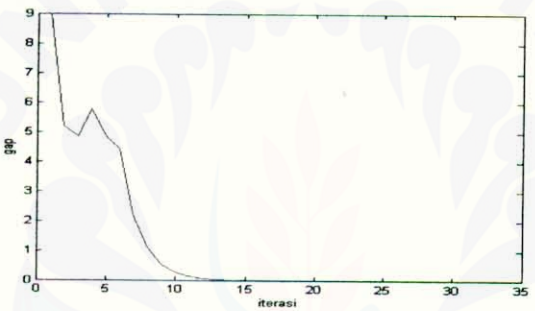
1. Untuk kapasitas saluran dengan matrik $p(y|x) = \begin{bmatrix} 0,3 & 0,2 & 0,5 \\ 0,5 & 0,3 & 0,2 \\ 0,2 & 0,5 & 0,3 \end{bmatrix}$

```
clear
clc
P=[0.3 0.2 0.5;0.5 0.3 0.2;0.2 0.5 0.3];
[N,M] = size(P);
gpvar z(M);
objektif = sum(z)
kendala = gpconstraint
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <=(prod( z.^(P(k,:)')));
end
A0=eye(N);
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1*(P(k,:))]
bi(k,:)=log([exp(-entropy(P(k,:)) )])
end
A=[A0;Ai]
b=[b0;bi]
szs=[size(A0,1);ones(N,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
```

Hasil program:

```
objektif is a posynomial with 3 monomial terms
objektif = 1*z_1^(1) + 1*z_2^(1) + 1*z_3^(1)
kendala is a constraint
A =
    1.0000         0         0
         0    1.0000         0
         0         0    1.0000
   -0.3000   -0.2000   -0.5000
   -0.5000   -0.3000   -0.2000
   -0.2000   -0.5000   -0.3000
b =
         0
         0
         0
   -1.4855
   -1.4855
   -1.4855
Iteration     primal obj.          gap         dual residual      previous step.
   1         1.00000e+000     9.00000e+000       7.34e-001             Inf
   2        -8.00503e+001     7.69348e+000       7.23e-001         1.56e-002
Iteration     primal obj.          gap         dual residual      previous step.
   1         8.21525e+001     9.00000e+000       5.61e-001             Inf
   2         2.43558e+000     5.20246e+000       5.35e-001         4.63e-002
   3         1.01555e-001     4.84410e+000       4.68e-001         1.25e-001
   4         2.81159e+000     5.77152e+000       2.34e-001         5.00e-001
   5         1.38093e+000     4.84555e+000       1.17e-001         5.00e-001
   6         1.04481e+000     4.43182e+000       5.00e-017         1.00e+000
   7         3.51819e-001     2.21591e+000       7.09e-017         1.00e+000
   8        -1.75446e-002     1.10795e+000       9.22e-017         1.00e+000
   9        -2.02204e-001     5.53977e-001       1.51e-016         1.00e+000
  10        -2.94533e-001     2.76989e-001       1.71e-016         1.00e+000
  11        -3.40698e-001     1.38494e-001       1.21e-017         1.00e+000
  12        -3.63781e-001     6.92472e-002       5.42e-017         1.00e+000
  13        -3.75322e-001     3.46236e-002       5.98e-017         1.00e+000
  14        -3.81092e-001     1.73118e-002       1.23e-016         1.00e+000
  15        -3.83978e-001     8.65589e-003       7.57e-017         1.00e+000
  16        -3.85420e-001     4.32795e-003       1.44e-016         1.00e+000
  17        -3.86142e-001     2.16397e-003       2.72e-017         1.00e+000
  18        -3.86502e-001     1.08199e-003       1.40e-016         1.00e+000
  19        -3.86683e-001     5.40993e-004       1.77e-016         1.00e+000
  20        -3.86773e-001     2.70497e-004       7.27e-017         1.00e+000
  21        -3.86818e-001     1.35248e-004       5.10e-017         1.00e+000
  22        -3.86840e-001     6.76242e-005       1.51e-016         1.00e+000
  23        -3.86852e-001     3.38121e-005       5.39e-017         1.00e+000
```

```
   24        -3.86857e-001      1.69060e-005      1.12e-016      1.00e+000
   25        -3.86860e-001      8.45302e-006      1.50e-016      1.00e+000
   26        -3.86862e-001      4.22651e-006      1.14e-016      1.00e+000
   27        -3.86862e-001      2.11326e-006      9.60e-017      1.00e+000
   28        -3.86863e-001      1.05663e-006      1.06e-016      1.00e+000
   29        -3.86863e-001      5.28314e-007      7.45e-017      1.00e+000
   30        -3.86863e-001      2.64157e-007      1.24e-016      1.00e+000
   31        -3.86863e-001      1.32078e-007      3.40e-017      1.00e+000
   32        -3.86863e-001      6.60392e-008      1.08e-016      1.00e+000
   33        -3.86863e-001      3.30196e-008      7.36e-017      1.00e+000
   34        -3.86863e-001      1.65098e-008      1.11e-016      1.00e+000
   35        -3.86863e-001      8.25490e-009      9.20e-017      1.00e+000
Solved
ans =
    0.6792
Plot Gambar 3.1
iterasi=1:35
gap=[9.00000e+000          5.20246e+000          4.84410e+000          5.77154e+000
4.84555e+000          4.43182e+000          2.21591e+000          1.10795e+000
5.53977e-001          2.76989e-001      1.38494e-001          6.92471e-002
3.46236e-002          1.73118e-002          8.65589e-003          4.32795e-003
2.16397e-003          1.08199e-003          5.40993e-004          2.70497e-004
1.35248e-004          6.76242e-005          3.38121e-005          1.69060e-005
8.45302e-006          4.22651e-006          2.11325e-006          1.05663e-006
5.28314e-007          2.64157e-007          1.32078e-007          6.60392e-008
3.30196e-008      1.65098e-008      8.25490e-009 ];
>> plot(iterasi,gap)
>> xlabel('iterasi');ylabel('gap');
```



2. $P = \begin{bmatrix} 0.4 & 0.4 & 0.2 \\ 0.2 & 0.4 & 0.4 \\ 0.4 & 0.2 & 0.4 \end{bmatrix}$

```
clear
clc
P=[0.4 0.4 0.2;0.2 0.4 0.4;0.4 0.2 0.4];
[N,M] = size(P);
gpvar z(M);
objektif = sum(z)
kendala = gpconstraint
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <=(prod( z.^(P(k,:)')));
end
A0=eye(N);
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1*(P(k,:))]
bi(k,:)=log([exp(-entropy(P(k,:)) )])
end
A=[A0;Ai]
b=[b0;bi]
szs=[size(A0,1);ones(N,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
objektif is a posynomial with 3 monomial terms
objektif = 1*z_1^(1) + 1*z_2^(1) + 1*z_3^(1)
 kendala is a constraint
A =
    1.0000         0         0
```

```
         0    1.0000         0
         0         0    1.0000
   -0.4000   -0.4000   -0.2000
   -0.2000   -0.4000   -0.4000
   -0.4000   -0.2000   -0.4000
b =
         0
         0
         0
   -1.5219
   -1.5219
   -1.5219
Iteration     primal obj.        gap          dual residual     previous step.
    1         1.00000e+000    9.00000e+000       7.19e-001           Inf
    2        -8.00504e+001    7.69349e+000       7.08e-001        1.56e-002
Iteration     primal obj.        gap          dual residual     previous step.
    1         8.21524e+001    9.00000e+000       5.61e-001           Inf
    2         2.43290e+000    5.20476e+000       5.35e-001        4.63e-002
    3         3.78544e-002    4.83903e+000       4.68e-001        1.25e-001
    4         2.82157e+000    5.79894e+000       2.34e-001        5.00e-001
    5         1.35176e+000    4.85130e+000       1.17e-001        5.00e-001
    6         1.00650e+000    4.42996e+000       8.43e-017        1.00e+000
    7         3.15057e-001    2.21498e+000       4.95e-017        1.00e+000
    8        -5.41523e-002    1.10749e+000       1.12e-016        1.00e+000
    9        -2.38734e-001    5.53745e-001       7.94e-017        1.00e+000
   10        -3.31025e-001    2.76872e-001       6.45e-017        1.00e+000
   11        -3.77170e-001    1.38436e-001       1.62e-016        1.00e+000
   12        -4.00243e-001    6.92181e-002       2.09e-016        1.00e+000
   13        -4.11779e-001    3.46091e-002       1.13e-016        1.00e+000
   14        -4.17548e-001    1.73045e-002       6.44e-017        1.00e+000
   15        -4.20432e-001    8.65226e-003       5.44e-017        1.00e+000
   16        -4.21874e-001    4.32613e-003       4.67e-017        1.00e+000
   17        -4.22595e-001    2.16307e-003       1.89e-016        1.00e+000
   18        -4.22955e-001    1.08153e-003       4.29e-017        1.00e+000
   19        -4.23136e-001    5.40767e-004       1.02e-016        1.00e+000
   20        -4.23226e-001    2.70383e-004       4.61e-017        1.00e+000
   21        -4.23271e-001    1.35192e-004       1.76e-016        1.00e+000
   22        -4.23293e-001    6.75958e-005       1.13e-016        1.00e+000
   23        -4.23305e-001    3.37979e-005       3.78e-017        1.00e+000
   24        -4.23310e-001    1.68990e-005       9.00e-017        1.00e+000
   25        -4.23313e-001    8.44948e-006       7.85e-017        1.00e+000
   26        -4.23314e-001    4.22474e-006       7.00e-017        1.00e+000
   27        -4.23315e-001    2.11237e-006       1.12e-016        1.00e+000
   28        -4.23315e-001    1.05618e-006       1.37e-016        1.00e+000
   29        -4.23316e-001    5.28092e-007       1.03e-016        1.00e+000
   30        -4.23316e-001    2.64046e-007       1.85e-016        1.00e+000
   31        -4.23316e-001    1.32023e-007       6.53e-017        1.00e+000
   32        -4.23316e-001    6.60115e-008       5.30e-017        1.00e+000
   33        -4.23316e-001    3.30058e-008       1.31e-016        1.00e+000
   34        -4.23316e-001    1.65029e-008       4.33e-017        1.00e+000
   35        -4.23316e-001    8.25144e-009       8.33e-017        1.00e+000
Solved
ans =
    0.6549
```

3. $P = \begin{bmatrix} 0{,}25 & 0{,}25 & 0{,}2 & 0{,}15 & 0{,}15 \\ 0{,}15 & 0{,}25 & 0{,}25 & 0{,}2 & 0{,}15 \\ 0{,}15 & 0{,}15 & 0{,}25 & 0{,}25 & 0{,}2 \\ 0{,}2 & 0{,}15 & 0{,}15 & 0{,}25 & 0{,}25 \\ 0{,}25 & 0{,}2 & 0{,}15 & 0{,}15 & 0{,}25 \end{bmatrix}$

```
clear
clc
P=[0.25 0.25 0.2 0.15 0.15;0.15 0.25 0.25 0.2 0.15;0.15 0.15 0.25 0.25 0.2;0.2 0.15 0.15 0.25
0.25;0.25 0.2 0.15 0.15 0.25];
[N,M] = size(P);
gpvar z(M);
objektif = sum(z)
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <=(prod( z.^(P(k,:)')))
end
```

```
A0=eye(N);
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1*(P(k,:))];
bi(k,:)=log([exp(-entropy(P(k,:)) )]);
end
A=[A0;Ai]
b=[b0;bi]
szs=[size(A0,1);ones(N,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
objektif is a posynomial with 5 monomial terms
 objektif = 1*z_1^(1) + 1*z_2^(1) + 1*z_3^(1) + 1*z_4^(1) + 1*z_5^(1)
 kendala is a constraint 0.10173 <= 1*z_1^(0.25)*z_2^(0.25)*z_3^(0.2)*z_4^(0.15)*z_5^(0.15)
kendala is a set of 5 constraints (index to get individual constraints).
A =
    1.0000         0         0         0         0
         0    1.0000         0         0         0
         0         0    1.0000         0         0
         0         0         0    1.0000         0
         0         0         0         0    1.0000
   -0.2500   -0.2500   -0.2000   -0.1500   -0.1500
   -0.1500   -0.2500   -0.2500   -0.2000   -0.1500
   -0.1500   -0.1500   -0.2500   -0.2500   -0.2000
   -0.2000   -0.1500   -0.1500   -0.2500   -0.2500
   -0.2500   -0.2000   -0.1500   -0.1500   -0.2500
b =

         0
         0
         0
         0
         0
   -2.2855
   -2.2855
   -2.2855
   -2.2855
   -2.2855

Iteration     primal obj.          gap          dual residual      previous step.
    1        1.00000e+000     1.50000e+001        8.82e-001             Inf
    2       -9.54690e+001     1.18702e+001        8.55e-001          3.13e-002
Iteration     primal obj.          gap          dual residual      previous step.

    1        9.80967e+001     1.50000e+001        4.33e-001             Inf
    2        2.63950e+001     1.07064e+001        4.09e-001          5.59e-002
    3        1.53916e+001     1.01756e+001        3.92e-001          4.00e-002
    4        8.50283e+000     9.63078e+000        3.68e-001          6.25e-002
    5        3.76690e+000     8.87776e+000        3.22e-001          1.25e-001
    6        2.75332e-001     6.93308e+000        1.61e-001          5.00e-001
    7        2.70032e+000     8.37219e+000        1.76e-016          1.00e+000
    8        7.19972e-001     4.18609e+000        1.57e-016          1.00e+000
    9        2.16397e-002     2.09305e+000        6.58e-017          1.00e+000
   10       -3.27196e-001     1.04652e+000        6.92e-017          1.00e+000
   11       -5.01617e-001     5.23262e-001        3.30e-017          1.00e+000
   12       -5.88827e-001     2.61631e-001        5.64e-017          1.00e+000
   13       -6.32432e-001     1.30815e-001        6.93e-017          1.00e+000
   14       -6.54235e-001     6.54077e-002        1.02e-016          1.00e+000
   15       -6.65136e-001     3.27039e-002        1.13e-016          1.00e+000
   16       -6.70587e-001     1.63519e-002        5.38e-017          1.00e+000
   17       -6.73312e-001     8.17597e-003        1.26e-016          1.00e+000
   18       -6.74675e-001     4.08798e-003        1.07e-016          1.00e+000
   19       -6.75356e-001     2.04399e-003        4.21e-017          1.00e+000
   20       -6.75697e-001     1.02200e-003        8.58e-017          1.00e+000
   21       -6.75867e-001     5.10998e-004        4.89e-017          1.00e+000
   22       -6.75952e-001     2.55499e-004        1.20e-016          1.00e+000
   23       -6.75995e-001     1.27749e-004        1.20e-016          1.00e+000
   24       -6.76016e-001     6.38747e-005        9.43e-017          1.00e+000
   25       -6.76027e-001     3.19374e-005        4.38e-017          1.00e+000
   26       -6.76032e-001     1.59687e-005        4.55e-017          1.00e+000
   27       -6.76035e-001     7.98434e-006        8.45e-017          1.00e+000
   28       -6.76036e-001     3.99217e-006        5.14e-017          1.00e+000
   29       -6.76037e-001     1.99609e-006        5.85e-017          1.00e+000
   30       -6.76037e-001     9.98043e-007        5.52e-017          1.00e+000
   31       -6.76037e-001     4.99021e-007        5.44e-017          1.00e+000
```

```
    32          -6.76037e-001       2.49511e-007       5.41e-017          1.00e+000
    33          -6.76037e-001       1.24755e-007       6.25e-017          1.00e+000
    34          -6.76037e-001       6.23777e-008       5.27e-017          1.00e+000
    35          -6.76037e-001       3.11888e-008       5.39e-017          1.00e+000
    36          -6.76037e-001       1.55944e-008       4.24e-017          1.00e+000
    37          -6.76037e-001       7.79721e-009       5.20e-017          1.00e+000
Solved
ans =
    0.5086
```

4. $P = \begin{bmatrix} 0,3 & 0,2 & 0,1 & 0,15 & 0,25 \\ 0,25 & 0,3 & 0,2 & 0,1 & 0,15 \\ 0,15 & 0,25 & 0,3 & 0,2 & 0,1 \\ 0,1 & 0,15 & 0,25 & 0,3 & 0,2 \\ 0,2 & 0,1 & 0,15 & 0,25 & 0,3 \end{bmatrix}$

```
clear
clc
P=[0.3 0.2 0.1 0.15 0.25; 0.25 0.3 0.2 0.1 0.15 ;0.15 0.25 0.3 0.2 0.1;0.1 0.15 0.25 0.3
0.2;0.2 0.1 0.15 0.25 0.3];
[N,M] = size(P);
gpvar z(M);
objektif = sum(z)
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <=(prod( z.^(P(k,:)')))
end
A0=eye(N);
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1*(P(k,:))];
bi(k,:)=log([exp(-entropy(P(k,:)) )]);
end
A=[A0;Ai]
b=[b0;bi]
szs=[size(A0,1);ones(N,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
```

hasil program

```
objektif is a posynomial with 5 monomial terms
 objektif = 1*z_1^(1) + 1*z_2^(1) + 1*z_3^(1) + 1*z_4^(1) + 1*z_5^(1)
 kendala is a constraint 0.10772 <= 1*z_1^(0.3)*z_2^(0.2)*z_3^(0.1)*z_4^(0.15)*z_5^(0.25)
 kendala is a set of 5 constraints (index to get individual constraints).

A =

    1.0000         0         0         0         0
         0    1.0000         0         0         0
         0         0    1.0000         0         0
         0         0         0    1.0000         0
         0         0         0         0    1.0000
   -0.3000   -0.2000   -0.1000   -0.1500   -0.2500
   -0.2500   -0.3000   -0.2000   -0.1000   -0.1500
   -0.1500   -0.2500   -0.3000   -0.2000   -0.1000
   -0.1000   -0.1500   -0.2500   -0.3000   -0.2000
   -0.2000   -0.1000   -0.1500   -0.2500   -0.3000

b =

         0
         0
         0
         0
         0
   -2.2282
   -2.2282
   -2.2282
   -2.2282
   -2.2282
Iteration     primal obj.          gap         dual residual      previous step.
```

```
    1        1.00000e+000     1.50000e+001     9.09e-001          Inf
    2       -9.54689e+001     1.18702e+001     8.81e-001     3.13e-002
 Iteration      primal obj.          gap       dual residual   previous step.

    1        9.80968e+001     1.50000e+001     4.33e-001          Inf
    2        2.63984e+001     1.07035e+001     4.08e-001     5.59e-002
    3        1.54017e+001     1.01718e+001     3.92e-001     4.01e-002
    4        8.54484e+000     9.62911e+000     3.68e-001     6.25e-002
    5        3.82352e+000     8.87788e+000     3.22e-001     1.25e-001
    6        3.34686e-001     6.93433e+000     1.61e-001     5.00e-001
    7        2.75613e+000     8.37077e+000     1.67e-016     1.00e+000
    8        7.76998e-001     4.18538e+000     9.89e-017     1.00e+000
    9        7.87838e-002     2.09269e+000     7.39e-017     1.00e+000
   10       -2.69993e-001     1.04635e+000     3.58e-017     1.00e+000
   11       -4.44384e-001     5.23173e-001     1.28e-016     1.00e+000
   12       -5.31580e-001     2.61587e-001     5.02e-017     1.00e+000
   13       -5.75177e-001     1.30793e-001     5.85e-017     1.00e+000
   14       -5.96976e-001     6.53966e-002     1.59e-016     1.00e+000
   15       -6.07876e-001     3.26983e-002     9.85e-017     1.00e+000
   16       -6.13325e-001     1.63492e-002     6.50e-017     1.00e+000
   17       -6.16050e-001     8.17458e-003     5.52e-017     1.00e+000
   18       -6.17413e-001     4.08729e-003     5.25e-017     1.00e+000
   19       -6.18094e-001     2.04364e-003     3.19e-017     1.00e+000
   20       -6.18434e-001     1.02182e-003     9.98e-017     1.00e+000
   21       -6.18605e-001     5.10911e-004     1.75e-016     1.00e+000
   22       -6.18690e-001     2.55456e-004     8.09e-017     1.00e+000
   23       -6.18732e-001     1.27728e-004     1.28e-016     1.00e+000
   24       -6.18754e-001     6.38639e-005     5.01e-017     1.00e+000
   25       -6.18764e-001     3.19320e-005     1.00e-016     1.00e+000
   26       -6.18770e-001     1.59660e-005     8.77e-017     1.00e+000
   27       -6.18772e-001     7.98299e-006     5.76e-017     1.00e+000
   28       -6.18774e-001     3.99149e-006     9.43e-017     1.00e+000
   29       -6.18774e-001     1.99575e-006     4.82e-017     1.00e+000
   30       -6.18775e-001     9.97873e-007     5.56e-017     1.00e+000
   31       -6.18775e-001     4.98937e-007     4.45e-017     1.00e+000
   32       -6.18775e-001     2.49468e-007     5.34e-017     1.00e+000
   33       -6.18775e-001     1.24734e-007     1.59e-017     1.00e+000
   34       -6.18775e-001     6.23671e-008     8.48e-017     1.00e+000
   35       -6.18775e-001     3.11835e-008     4.86e-017     1.00e+000
   36       -6.18775e-001     1.55918e-008     4.31e-017     1.00e+000
   37       -6.18775e-001     7.79589e-009     9.20e-017     1.00e+000
Solved
ans =
    0.5386
```

5. $P = \begin{bmatrix} 0,7 & 0,15 & 0,05 & 0,05 & 0,05 \\ 0,05 & 0,7 & 0,15 & 0,05 & 0,05 \\ 0,05 & 0,05 & 0,7 & 0,15 & 0,05 \\ 0,05 & 0,05 & 0,05 & 0,7 & 0,15 \\ 0,15 & 0,05 & 0,05 & 0,05 & 0,7 \end{bmatrix}$

```
clear
clc
P=[0.7 0.15 0.05 0.05 0.05; 0.05 0.7 0.015 0.05 0.05 ;0.05 0.05 0.7 0.15 0.05;0.05 0.05 0.05
0.7 0.15;0.15 0.05 0.05 0.05 0.7];
 [N,M] = size(P);
gpvar z(M);
objektif = sum(z)
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <=(prod( z.^(P(k,:)')))
end
A0=eye(N);
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1*(P(k,:))];
bi(k,:)=log([exp(-entropy(P(k,:)) )]);
end
A=[A0;Ai]
b=[b0;bi]
szs=[size(A0,1);ones(N,1)]
 [x,status]=gpcvx(A,b,szs);
```

```
z=exp(x);
sum(z)
```

hasil program

```
objektif is a posynomial with 5 monomial terms
 objektif = 1*z_1^(1) + 1*z_2^(1) + 1*z_3^(1) + 1*z_4^(1) + 1*z_5^(1)
 kendala is a constraint 0.24195 <= 1*z_1^(0.7)*z_2^(0.15)*z_3^(0.05)*z_4^(0.05)*z_5^(0.05)
kendala is a set of 5 constraints (index to get individual constraints).

A =

    1.0000         0         0         0         0
         0    1.0000         0         0         0
         0         0    1.0000         0         0
         0         0         0    1.0000         0
         0         0         0         0    1.0000
   -0.7000   -0.1500   -0.0500   -0.0500   -0.0500
   -0.0500   -0.7000   -0.0150   -0.0500   -0.0500
   -0.0500   -0.0500   -0.7000   -0.1500   -0.0500
   -0.0500   -0.0500   -0.0500   -0.7000   -0.1500
   -0.1500   -0.0500   -0.0500   -0.0500   -0.7000

b =
         0
         0
         0
         0
         0
   -1.4190
   -1.0994
   -1.4190
   -1.4190
   -1.4190
```

Iteration	primal obj.	gap	dual residual	previous step.
1	1.00000e+000	1.50000e+001	1.49e+000	Inf
2	-6.39760e+001	1.34830e+001	1.46e+000	2.17e-002

Iteration	primal obj.	gap	dual residual	previous step.
1	7.77836e+001	1.50000e+001	9.88e-001	Inf
2	7.92262e+001	1.38292e+001	9.87e-001	1.33e-002
3	8.32810e+001	1.21726e+001	9.90e-001	2.63e-002
4	7.39322e+001	1.11371e+001	9.85e-001	1.22e-002
5	3.86109e+001	1.07727e+001	9.80e-001	7.89e-003
6	2.04638e+001	1.04432e+001	9.69e-001	1.57e-002
7	1.44961e+001	1.00373e+001	6.12e-001	2.99e-002
8	1.00929e+001	9.92728e+000	5.46e-001	1.76e-002
9	4.90649e+000	9.11556e+000	4.67e-001	1.25e-001
10	1.23050e+000	7.09076e+000	2.04e-001	5.00e-001
11	3.59783e+000	8.38190e+000	2.56e-002	1.00e+000
12	1.62040e+000	4.19674e+000	1.89e-003	1.00e+000
13	9.19193e-001	2.09982e+000	2.12e-004	1.00e+000
14	5.68918e-001	1.05017e+000	6.50e-005	1.00e+000
15	3.94151e-001	5.25213e-001	3.40e-005	1.00e+000
16	3.06797e-001	2.62700e-001	1.67e-005	1.00e+000
17	2.63056e-001	1.31390e-001	6.15e-006	1.00e+000
18	2.41162e-001	6.57070e-002	1.78e-006	1.00e+000
19	2.30210e-001	3.28566e-002	4.61e-007	1.00e+000
20	2.24734e-001	1.64291e-002	1.16e-007	1.00e+000
21	2.21996e-001	8.21475e-003	2.90e-008	1.00e+000
22	2.20627e-001	4.10742e-003	7.24e-009	1.00e+000
23	2.19942e-001	2.05372e-003	1.81e-009	1.00e+000
24	2.19600e-001	1.02687e-003	4.52e-010	1.00e+000
25	2.19429e-001	5.13433e-004	1.13e-010	1.00e+000
26	2.19343e-001	2.56717e-004	2.83e-011	1.00e+000
27	2.19300e-001	1.28359e-004	7.07e-012	1.00e+000
28	2.19279e-001	6.41793e-005	1.77e-012	1.00e+000
29	2.19268e-001	3.20896e-005	4.42e-013	1.00e+000
30	2.19263e-001	1.60448e-005	1.11e-013	1.00e+000
31	2.19260e-001	8.02241e-006	2.76e-014	1.00e+000
32	2.19259e-001	4.01120e-006	6.92e-015	1.00e+000
33	2.19258e-001	2.00560e-006	1.73e-015	1.00e+000
34	2.19258e-001	1.00280e-006	4.40e-016	1.00e+000
35	2.19258e-001	5.01401e-007	1.23e-016	1.00e+000
36	2.19258e-001	2.50700e-007	4.40e-017	1.00e+000
37	2.19258e-001	1.25350e-007	9.14e-017	1.00e+000
38	2.19258e-001	6.26751e-008	1.99e-016	1.00e+000
39	2.19258e-001	3.13375e-008	7.85e-017	1.00e+000

```
  40          2.19258e-001      1.56688e-008      8.24e-017         1.00e+000
  41          2.19258e-001      7.83438e-009      3.57e-017         1.00e+000
Solved
ans =
    1.2452
```

6. $P = \begin{bmatrix} 0{,}02 & 0{,}02 & 0{,}02 & \cdots & 0{,}02 \\ 0{,}02 & 0{,}02 & 0{,}02 & \cdots & 0{,}02 \\ 0{,}02 & 0{,}02 & 0{,}02 & \cdots & 0{,}02 \cdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0{,}02 & 0{,}02 & 0{,}02 & \cdots & 0{,}02 \end{bmatrix}$, P50x50

```
clear
clc
A=1/50;
P=A*ones(50);
[N,M] = size(P);
gpvar z(M);
objektif = sum(z)
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <=(prod( z.^(P(k,:)')))
end
A0=eye(N);
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1*(P(k,:))];
bi(k,:)=log([exp(-entropy(P(k,:)) )]);
end
A=[A0;Ai]
b=[b0;bi]
szs=[size(A0,1);ones(N,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
hasil program
Iteration     primal obj.          gap         dual residual     previous step.
   1         1.00000e+000     1.50000e+002        7.01e+000             Inf
   2        -6.16930e+001     1.20426e+002        5.25e+000         2.50e-001
Iteration     primal obj.          gap         dual residual     previous step.
   1         6.73466e+001     1.50000e+002        5.44e-002             Inf
   2         1.16195e+001     5.35638e+001        1.81e-014         1.00e+000
   3         6.95326e+000     2.67819e+001        2.37e-016         1.00e+000
   4         2.73531e+000     1.33909e+001        1.74e-015         1.00e+000
   5         5.00034e-001     6.69547e+000        1.75e-016         1.00e+000
   6        -6.15921e-001     3.34774e+000        1.31e-015         1.00e+000
   7        -1.17388e+000     1.67387e+000        1.43e-016         1.00e+000
   8        -1.45286e+000     8.36934e-001        7.57e-017         1.00e+000
   9        -1.59234e+000     4.18467e-001        2.16e-016         1.00e+000
  10        -1.66209e+000     2.09234e-001        4.63e-016         1.00e+000
  11        -1.69696e+000     1.04617e-001        6.27e-017         1.00e+000
  12        -1.71440e+000     5.23084e-002        5.70e-016         1.00e+000
  13        -1.72312e+000     2.61542e-002        3.77e-016         1.00e+000
  14        -1.72747e+000     1.30771e-002        4.03e-016         1.00e+000
  15        -1.72965e+000     6.53855e-003        2.08e-016         1.00e+000
  16        -1.73074e+000     3.26927e-003        2.87e-016         1.00e+000
  17        -1.73129e+000     1.63464e-003        2.54e-016         1.00e+000
  18        -1.73156e+000     8.17319e-004        1.21e-016         1.00e+000
  19        -1.73170e+000     4.08659e-004        1.06e-016         1.00e+000
  20        -1.73177e+000     2.04330e-004        2.71e-017         1.00e+000
  21        -1.73180e+000     1.02165e-004        4.01e-016         1.00e+000
  22        -1.73182e+000     5.10824e-005        1.24e-016         1.00e+000
  23        -1.73182e+000     2.55412e-005        4.45e-017         1.00e+000
  24        -1.73183e+000     1.27706e-005        6.83e-016         1.00e+000
  25        -1.73183e+000     6.38530e-006        4.27e-016         1.00e+000
  26        -1.73183e+000     3.19265e-006        3.26e-016         1.00e+000
  27        -1.73183e+000     1.59633e-006        1.57e-016         1.00e+000
  28        -1.73183e+000     7.98163e-007        1.07e-016         1.00e+000
  29        -1.73183e+000     3.99081e-007        5.18e-016         1.00e+000
  30        -1.73183e+000     1.99541e-007        1.20e-016         1.00e+000
  31        -1.73183e+000     9.97703e-008        1.32e-016         1.00e+000
  32        -1.73183e+000     4.98852e-008        6.03e-017         1.00e+000
```

```
  33        -1.73183e+000     2.49426e-008     7.00e-016        1.00e+000
  34        -1.73183e+000     1.24713e-008     1.61e-016        1.00e+000
  35        -1.73183e+000     6.23564e-009     3.68e-016        1.00e+000
Solved
ans =
    0.1770
```

7. $P = \begin{bmatrix} 0{,}01 & 0{,}01 & 0{,}01 & \cdots & 0{,}01 \\ 0{,}01 & 0{,}01 & 0{,}01 & \cdots & 0{,}01 \\ 0{,}01 & 0{,}01 & 0{,}01 & \cdots & 0{,}01 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0{,}01 & 0{,}01 & 0{,}01 & \cdots & 0{,}01 \end{bmatrix}$, P100x100

```
clear
clc
P=0.01*ones(100);
[N,M] = size(P);
gpvar z(M);
objektif = sum(z)
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <=(prod( z.^(P(k,:)')))
end
A0=eye(N);
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1*(P(k,:))];
bi(k,:)=log([exp(-entropy(P(k,:)) )]);
end
A=[A0;Ai]
b=[b0;bi]
szs=[size(A0,1);ones(N,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
```

hasil program

```
objektif is a posynomial with 100 monomial terms
objektif = 1*z_1^(1) + 1*z_2^(1) + 1*z_3^(1) + … 1*z_100^(1)
kendala is a constraint 0.001302 <= 1*z_1^(0.01)*z_2^(0.01)*z_3^(0.01)*…*z_100^(0.01)
kendala is a set of 100 constraints (index to get individual constraints).
Iteration     primal obj.          gap         dual residual      previous step.
   1         1.00000e+000     3.00000e+002      1.29e+001             Inf
   2        -9.19311e+001     8.34610e+001      5.96e-012          1.00e+000
Iteration     primal obj.          gap         dual residual      previous step.
   1         1.01179e+002     3.00000e+002      3.94e-002             Inf
   2         5.29859e+001     1.30992e+002      7.39e-014          1.00e+000
   3         1.79953e+001     6.54959e+001      8.02e-015          1.00e+000
   4         8.84794e+000     3.27480e+001      2.12e-015          1.00e+000
   5         3.42120e+000     1.63740e+001      1.51e-015          1.00e+000
   6         6.90322e-001     8.18699e+000      2.23e-015          1.00e+000
   7        -6.74185e-001     4.09350e+000      1.82e-015          1.00e+000
   8        -1.35643e+000     2.04675e+000      6.29e-016          1.00e+000
   9        -1.69756e+000     1.02337e+000      5.15e-016          1.00e+000
  10        -1.86812e+000     5.11687e-001      1.85e-016          1.00e+000
  11        -1.95340e+000     2.55844e-001      1.35e-016          1.00e+000
  12        -1.99604e+000     1.27922e-001      4.92e-016          1.00e+000
  13        -2.01736e+000     6.39609e-002      5.98e-016          1.00e+000
  14        -2.02802e+000     3.19804e-002      2.66e-016          1.00e+000
  15        -2.03335e+000     1.59902e-002      1.26e-016          1.00e+000
  16        -2.03602e+000     7.99511e-003      3.38e-016          1.00e+000
  17        -2.03735e+000     3.99756e-003      6.73e-016          1.00e+000
  18        -2.03802e+000     1.99878e-003      1.50e-016          1.00e+000
  19        -2.03835e+000     9.99389e-004      2.72e-016          1.00e+000
  20        -2.03852e+000     4.99694e-004      1.66e-016          1.00e+000
  21        -2.03860e+000     2.49847e-004      6.61e-016          1.00e+000
  22        -2.03864e+000     1.24924e-004      3.05e-016          1.00e+000
  23        -2.03866e+000     6.24618e-005      5.92e-017          1.00e+000
  24        -2.03867e+000     3.12309e-005      1.74e-016          1.00e+000
  25        -2.03868e+000     1.56154e-005      2.51e-016          1.00e+000
  26        -2.03868e+000     7.80772e-006      3.80e-016          1.00e+000
  27        -2.03868e+000     3.90386e-006      5.07e-016          1.00e+000
  28        -2.03868e+000     1.95193e-006      5.51e-016          1.00e+000
```

```
   29        -2.03868e+000      9.75966e-007      6.17e-016      1.00e+000
   30        -2.03868e+000      4.87983e-007      8.06e-016      1.00e+000
   31        -2.03868e+000      2.43991e-007      8.27e-016      1.00e+000
   32        -2.03868e+000      1.21996e-007      7.40e-016      1.00e+000
   33        -2.03868e+000      6.09979e-008      1.36e-016      1.00e+000
   34        -2.03868e+000      3.04989e-008      1.80e-016      1.00e+000
   35        -2.03868e+000      1.52495e-008      1.43e-016      1.00e+000
   36        -2.03868e+000      7.62472e-009      2.00e-016      1.00e+000
status =
Solved
ans =
    0.1302
```

8. $P = \begin{bmatrix} 0{,}005 & 0{,}005 & 0{,}005 & \cdots & 0{,}005 \\ 0{,}005 & 0{,}005 & 0{,}005 & \cdots & 0{,}005 \\ 0{,}005 & 0{,}005 & 0{,}005 & \cdots & 0{,}005 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0{,}005 & 0{,}005 & 0{,}005 & \cdots & 0{,}005 \end{bmatrix}$, $P_{200x200}$

```
clear
clc
P=0.005*ones(200);
[N,M] = size(P);
gpvar z(M);
objektif = sum(z)
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <=(prod( z.^(P(k,:)')))
end
A0=eye(N);
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1*(P(k,:))];
bi(k,:)=log([exp(-entropy(P(k,:)) )]);
end
A=[A0;Ai]
b=[b0;bi]
szs=[size(A0,1);ones(N,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
```

hasil program

```
   1        1.00000e+000      6.00000e+002      2.38e+001            Inf
   2       -1.27918e+001      2.11652e+002      1.20e-001      9.95e-001
Iteration     primal obj.          gap        dual residual    previous step.
   1        2.33475e+001      6.00000e+002      4.72e-001            Inf
   2        3.25052e+001      2.38935e+002      1.63e-015      1.00e+000
   3        3.55432e+001      1.19468e+002      2.39e-015      1.00e+000
   4        1.84652e+001      5.97339e+001      1.48e-015      1.00e+000
   5        7.62706e+000      2.98669e+001      9.56e-015      1.00e+000
   6        2.63167e+000      1.49335e+001      2.23e-015      1.00e+000
   7        1.43373e-001      7.46673e+000      1.34e-015      1.00e+000
   8       -1.10108e+000      3.73337e+000      6.24e-016      1.00e+000
   9       -1.72331e+000      1.86668e+000      5.05e-016      1.00e+000
  10       -2.03442e+000      9.33342e-001      1.64e-016      1.00e+000
  11       -2.18998e+000      4.66671e-001      6.01e-016      1.00e+000
  12       -2.26776e+000      2.33335e-001      7.70e-016      1.00e+000
  13       -2.30665e+000      1.16668e-001      4.73e-016      1.00e+000
  14       -2.32609e+000      5.83338e-002      3.37e-016      1.00e+000
  15       -2.33581e+000      2.91669e-002      8.26e-017      1.00e+000
  16       -2.34067e+000      1.45835e-002      5.33e-016      1.00e+000
  17       -2.34310e+000      7.29173e-003      7.23e-016      1.00e+000
  18       -2.34432e+000      3.64587e-003      9.03e-016      1.00e+000
  19       -2.34493e+000      1.82293e-003      4.34e-016      1.00e+000
  20       -2.34523e+000      9.11466e-004      3.54e-016      1.00e+000
  21       -2.34538e+000      4.55733e-004      1.45e-016      1.00e+000
  22       -2.34546e+000      2.27867e-004      6.86e-016      1.00e+000
  23       -2.34550e+000      1.13933e-004      4.27e-016      1.00e+000
  24       -2.34552e+000      5.69666e-005      5.77e-016      1.00e+000
  25       -2.34552e+000      2.84833e-005      4.24e-016      1.00e+000
  26       -2.34553e+000      1.42417e-005      9.95e-016      1.00e+000
  27       -2.34553e+000      7.12083e-006      1.28e-017      1.00e+000
```

```
  28        -2.34553e+000     3.56042e-006     1.21e-016        1.00e+000
  29        -2.34553e+000     1.78021e-006     9.35e-016        1.00e+000
  30        -2.34553e+000     8.90104e-007     5.05e-016        1.00e+000
  31        -2.34553e+000     4.45052e-007     3.68e-016        1.00e+000
  32        -2.34553e+000     2.22526e-007     3.26e-016        1.00e+000
  33        -2.34553e+000     1.11263e-007     1.79e-016        1.00e+000
  34        -2.34553e+000     5.56315e-008     8.71e-016        1.00e+000
  35        -2.34553e+000     2.78158e-008     7.95e-016        1.00e+000
  36        -2.34553e+000     1.39078e-008     3.39e-018        1.00e+000
  37        -2.34553e+000     6.95398e-009     6.38e-016        1.00e+000
Solved
x =
   -7.6439
ans =
    0.0958
```

Kapasitas saluran dengan inputcost

1. $p(y|x) = \begin{bmatrix} 0,3 & 0,2 & 0,5 \\ 0,5 & 0,3 & 0,2 \\ 0,2 & 0,5 & 0,3 \end{bmatrix}$

```
clear
clc
P=[0.3 0.2 0.5;0.5 0.3 0.2;0.2 0.5 0.3];
S=1;
[N,M] = size(P);
si=[ones(N,1)];
% variabel GP
gpvar z(M) w
% fungsi objektif adalah channel capacity
obj = w^S*sum(z)
% kendala
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <= w^si(k)*(prod( z.^(P(k,:)')) )
end
A0=[ones(N,1) eye(N)];
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1 -1*(P(k,:))]
%A=(An)'
bi(k,:)=log([exp(-entropy(P(k,:)) )])
%szs= (szsl)'
end
Ain=[-1 zeros(1,N)]
bin=log([1])
A=[A0;Ai;Ain]
b=[b0;bi;bin]
szs=[size(A0,1);ones(N+1,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
```

hasil program

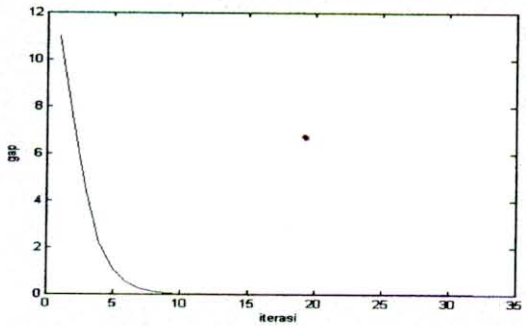
```
obj is a posynomial with 3 monomial terms

obj = 1*w^(1)*z_1^(1) + 1*w^(1)*z_2^(1) + 1*w^(1)*z_3^(1)

kendala is a set of 3 constraints (index to get individual constraints).
Ai =
   -1.0000   -0.3000   -0.2000   -0.5000
   -1.0000   -0.5000   -0.3000   -0.2000
   -1.0000   -0.2000   -0.5000   -0.3000
bi =
   -1.4855
   -1.4855
   -1.4855
Ain =
    -1     0     0     0
bin =
     0
A =
    1.0000    1.0000         0         0
    1.0000         0    1.0000         0
```

```
    1.0000         0         0    1.0000
   -1.0000   -0.3000   -0.2000   -0.5000
   -1.0000   -0.5000   -0.3000   -0.2000
   -1.0000   -0.2000   -0.5000   -0.3000
   -1.0000         0         0         0


b =
         0
         0
         0
   -1.4855
   -1.4855
   -1.4855
         0
szs =
     3
     1
     1
     1
     1
Iteration      primal obj.          gap          dual residual     previous step.
    1         1.00000e+000     1.20000e+001        2.63e+000             Inf
    2        -9.49078e+001     1.04704e+001        2.58e+000         1.55e-002
Iteration      primal obj.          gap          dual residual     previous step.
    1         9.70256e+001     1.20000e+001        1.11e+000             Inf
    2         5.59600e+000     8.70003e+000        1.07e+000         3.50e-002
    3         9.21579e-001     8.25829e+000        1.01e+000         6.25e-002
    4         2.60015e+000     7.40143e+000        4.00e-015         1.00e+000
    5         5.38186e-001     3.70071e+000        7.90e-016         1.00e+000
    6         7.57193e-002     1.85036e+000        4.53e-016         1.00e+000
    7        -1.55568e-001     9.25179e-001        7.19e-016         1.00e+000
    8        -2.71216e-001     4.62589e-001        8.38e-016         1.00e+000
    9        -3.29039e-001     2.31295e-001        1.24e-015         1.00e+000
   10        -3.57951e-001     1.15647e-001        1.86e-015         1.00e+000
   11        -3.72407e-001     5.78237e-002        9.34e-016         1.00e+000
   12        -3.79635e-001     2.89118e-002        1.52e-015         1.00e+000
   13        -3.83249e-001     1.44559e-002        1.22e-015         1.00e+000
   14        -3.85056e-001     7.22796e-003        2.45e-015         1.00e+000
   15        -3.85960e-001     3.61398e-003        1.66e-015         1.00e+000
   16        -3.86411e-001     1.80699e-003        1.47e-015         1.00e+000
   17        -3.86637e-001     9.03495e-004        2.25e-015         1.00e+000
   18        -3.86750e-001     4.51747e-004        1.09e-015         1.00e+000
   19        -3.86807e-001     2.25874e-004        8.27e-016         1.00e+000
   20        -3.86835e-001     1.12937e-004        1.91e-015         1.00e+000
   21        -3.86849e-001     5.64684e-005        2.57e-015         1.00e+000
   22        -3.86856e-001     2.82342e-005        1.33e-015         1.00e+000
   23        -3.86859e-001     1.41171e-005        8.73e-016         1.00e+000
   24        -3.86861e-001     7.05855e-006        8.70e-016         1.00e+000
   25        -3.86862e-001     3.52928e-006        7.40e-016         1.00e+000
   26        -3.86863e-001     1.76464e-006        3.27e-015         1.00e+000
   27        -3.86863e-001     8.82319e-007        2.63e-015         1.00e+000
   28        -3.86863e-001     4.41160e-007        5.40e-015         1.00e+000
   29        -3.86863e-001     2.20580e-007        2.88e-015         1.00e+000
   30        -3.86863e-001     1.10290e-007        2.66e-015         1.00e+000
   31        -3.86863e-001     5.51449e-008        1.36e-015         1.00e+000
   32        -3.86863e-001     2.75725e-008        8.49e-016         1.00e+000
   33        -3.86863e-001     1.37862e-008        6.70e-016         1.00e+000
   34        -3.86863e-001     6.89311e-009        2.68e-015         1.00e+000
Solved
ans =
  8.3680e+035
Plot gap 3.3
>> iterasi=1:32
>> gap=[     1.10000e+001          7.61167e+000          4.37554e+000          2.18777e+000
1.09389e+000          5.46943e-001          2.73472e-001          1.36736e-001
6.83679e-002          3.41840e-002          1.70920e-002          8.54599e-003
4.27299e-003          2.13650e-003          1.06825e-003          5.34124e-004
2.67062e-004          1.33531e-004          6.67655e-005          3.33828e-005
1.66914e-005          8.34569e-006          4.17285e-006          2.08642e-006
1.04321e-006          5.21606e-007          2.60803e-007          1.30401e-007
6.52007e-008    3.26004e-008    1.63002e-008    8.15010e-009    ];
>> plot(iterasi,gap)
>> xlabel('iterasi');ylabel('gap');
```



2. $P = \begin{bmatrix} 0.4 & 0.4 & 0.2 \\ 0.2 & 0.4 & 0.4 \\ 0.4 & 0.2 & 0.4 \end{bmatrix}$

```
clear
clc
P=[0.4 0.4 0.2;0.2 0.4 0.4;0.4 0.2 0.4];
S=1;
si=[1; 1 ;1];
% jumlah input (N) and output (M)
[N,M] = size(P);
% variabel GP
gpvar z(M) w
% fungsi objektif adalah channel capacity
obj = w^S*sum(z)
% kendala
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <= w^si(k)*(prod( z.^(P(k,:)')) )
end
A0=[ones(N,1) eye(N)];
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1 -1*(P(k,:))]
%A=(An)'
bi(k,:)=log([exp(-entropy(P(k,:)) )])
%szs= (szs1)'
end
Ain=[-1 zeros(1,N)]
bin=log([1])
A=[A0;Ai;Ain]
b=[b0;bi;bin]
szs=[size(A0,1);ones(N+1,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
hasil program
obj is a posynomial with 3 monomial terms
obj = 1*w^(1)*z_1^(1) + 1*w^(1)*z_2^(1) + 1*w^(1)*z_3^(1)
kendala is a constraint 0.21829 <= 1*w^(1)*z_1^(0.4)*z_2^(0.4)*z_3^(0.2)
kendala is a set of 2 constraints (index to get individual constraints).
kendala is a set of 3 constraints (index to get individual constraints).
Ai =

   -1.0000   -0.4000   -0.4000   -0.2000
   -1.0000   -0.2000   -0.4000   -0.4000
   -1.0000   -0.4000   -0.2000   -0.4000
bi =
   -1.5219
   -1.5219
   -1.5219
Ain =
    -1     0     0     0
bin =
     0
A =
    1.0000    1.0000         0         0
    1.0000         0    1.0000         0
```

```
    1.0000         0         0    1.0000
   -1.0000   -0.4000   -0.4000   -0.2000
   -1.0000   -0.2000   -0.4000   -0.4000
   -1.0000   -0.4000   -0.2000   -0.4000
   -1.0000         0         0         0
b =
         0
         0
         0
   -1.5219
   -1.5219
   -1.5219
         0
szs =
     3
     1
     1
     1
     1
Iteration     primal obj.          gap          dual residual      previous step.
    1         1.00000e+000     1.20000e+001        2.60e+000             Inf
    2        -9.49080e+001     1.04703e+001        2.56e+000         1.55e-002
Iteration     primal obj.          gap          dual residual      previous step.
    1         9.70261e+001     1.20000e+001        1.11e+000             Inf
    2         5.59193e+000     8.70180e+000        1.07e+000         3.49e-002
    3         8.81389e-001     8.25783e+000        1.01e+000         6.25e-002
    4         2.59213e+000     7.42965e+000        1.81e-015         1.00e+000
    5         5.05275e-001     3.71482e+000        3.20e-015         1.00e+000
    6         4.10302e-002     1.85741e+000        1.37e-015         1.00e+000
    7        -1.91139e-001     9.28706e-001        9.61e-016         1.00e+000
    8        -3.07228e-001     4.64353e-001        5.90e-016         1.00e+000
    9        -3.65272e-001     2.32177e-001        1.38e-015         1.00e+000
   10        -3.94294e-001     1.16088e-001        2.23e-015         1.00e+000
   11        -4.08805e-001     5.80441e-002        2.02e-015         1.00e+000
   12        -4.16060e-001     2.90221e-002        2.28e-016         1.00e+000
   13        -4.19688e-001     1.45110e-002        4.47e-016         1.00e+000
   14        -4.21502e-001     7.25552e-003        1.39e-015         1.00e+000
   15        -4.22409e-001     3.62776e-003        1.46e-015         1.00e+000
   16        -4.22862e-001     1.81388e-003        2.88e-015         1.00e+000
   17        -4.23089e-001     9.06940e-004        1.65e-015         1.00e+000
   18        -4.23202e-001     4.53470e-004        1.83e-015         1.00e+000
   19        -4.23259e-001     2.26735e-004        2.05e-015         1.00e+000
   20        -4.23287e-001     1.13367e-004        1.91e-015         1.00e+000
   21        -4.23302e-001     5.66837e-005        2.04e-015         1.00e+000
   22        -4.23309e-001     2.83419e-005        1.75e-015         1.00e+000
   23        -4.23312e-001     1.41709e-005        1.90e-015         1.00e+000
   24        -4.23314e-001     7.08547e-006        1.98e-015         1.00e+000
   25        -4.23315e-001     3.54273e-006        1.55e-015         1.00e+000
   26        -4.23315e-001     1.77137e-006        1.74e-015         1.00e+000
   27        -4.23316e-001     8.85683e-007        2.56e-015         1.00e+000
   28        -4.23316e-001     4.42842e-007        2.60e-015         1.00e+000
   29        -4.23316e-001     2.21421e-007        1.93e-015         1.00e+000
   30        -4.23316e-001     1.10710e-007        2.45e-015         1.00e+000
   31        -4.23316e-001     5.53552e-008        2.56e-015         1.00e+000
   32        -4.23316e-001     2.76776e-008        2.59e-015         1.00e+000
   33        -4.23316e-001     1.38388e-008        1.65e-015         1.00e+000
   34        -4.23316e-001     6.91940e-009        1.67e-015         1.00e+000
Solved
ans =
  8.2420e+035
```

3. $P = \begin{bmatrix} 0,3 & 0,2 & 0,1 & 0,15 & 0,25 \\ 0,25 & 0,3 & 0,2 & 0,1 & 0,15 \\ 0,15 & 0,25 & 0,3 & 0,2 & 0,1 \\ 0,1 & 0,15 & 0,25 & 0,3 & 0,2 \\ 0,2 & 0,1 & 0,15 & 0,25 & 0,3 \end{bmatrix}$

```
clear
clc
P=[0.3 0.2 0.1 0.15 0.25; 0.25 0.3 0.2 0.1 0.15 ;0.15 0.25 0.3 0.2 0.1;0.1 0.15 0.25 0.3
0.2;0.2 0.1 0.15 0.25 0.3]
[N,M] = size(P);
```

```
si=[ones(N,1)];
% variabel GP
gpvar z(M) w
% fungsi objektif adalah channel capacity
obj = w^S*sum(z)
% kendala
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <= w^si(k)*(prod( z.^(P(k,:)')) )
end
A0=[ones(N,1) eye(N)];
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1 -1*(P(k,:))]
%A=(An)'
bi(k,:)=log([exp(-entropy(P(k,:)) )])
%szs= (szs1)'
end
Ain=[-1 zeros(1,N)]
bin=log([1])
A=[A0;Ai;Ain]
b=[b0;bi;bin]
szs=[size(A0,1);ones(N+1,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
```

hasil program

```
objektif is a posynomial with 5 monomial terms
objektif   =   1*w^(1)*z_1^(1)   +   1*w^(1)*z_2^(1)   +   1*w^(1)*z_3^(1)   +   1*w^(1)*z_4^(1)   +
1*w^(1)*z_5^(1)
 kendala is a constraint 0.10772 <= 1*w^(1)*z_1^(0.3)*z_2^(0.2)*z_3^(0.1)*z_4^(0.15)*z_5^(0.25)
kendala is a set of 5 constraints (index to get individual constraints).
Iteration     primal obj.          gap          dual residual     previous step.
   1         1.00000e+000     1.80000e+001       3.72e+000            Inf
   2        -6.69506e+001     1.68592e+001       3.66e+000         1.55e-002
Iteration     primal obj.          gap          dual residual     previous step.
   1         6.95795e+001     1.80000e+001       1.01e+000            Inf
   2         2.99608e+001     1.58703e+001       9.58e-001         4.67e-002
   3         3.84880e+000     1.25880e+001       8.53e-001         1.10e-001
   4         8.36061e-001     9.58445e+000       4.27e-001         5.00e-001
   5         3.76334e+000     1.00893e+001       4.64e-015         1.00e+000
   6         1.48883e+000     5.04463e+000       6.99e-016         1.00e+000
   7         7.88137e-001     2.52231e+000       4.35e-016         1.00e+000
   8         4.37829e-001     1.26116e+000       2.40e-015         1.00e+000
   9         2.62668e-001     6.30579e-001       4.34e-016         1.00e+000
  10         1.75087e-001     3.15289e-001       7.83e-016         1.00e+000
  11         1.31297e-001     1.57645e-001       1.04e-015         1.00e+000
  12         1.09402e-001     7.88223e-002       2.10e-015         1.00e+000
  13         9.84546e-002     3.94112e-002       8.33e-016         1.00e+000
  14         9.29809e-002     1.97056e-002       1.33e-015         1.00e+000
  15         9.02440e-002     9.85279e-003       2.00e-015         1.00e+000
  16         8.88755e-002     4.92640e-003       8.99e-016         1.00e+000
  17         8.81913e-002     2.46320e-003       1.23e-015         1.00e+000
  18         8.78492e-002     1.23160e-003       2.52e-015         1.00e+000
  19         8.76781e-002     6.15799e-004       1.55e-015         1.00e+000
  20         8.75926e-002     3.07900e-004       1.29e-015         1.00e+000
  21         8.75499e-002     1.53950e-004       1.80e-015         1.00e+000
  22         8.75285e-002     7.69749e-005       1.69e-015         1.00e+000
  23         8.75178e-002     3.84875e-005       9.35e-016         1.00e+000
  24         8.75124e-002     1.92437e-005       1.01e-015         1.00e+000
  25         8.75098e-002     9.62187e-006       1.31e-015         1.00e+000
  26         8.75084e-002     4.81093e-006       1.91e-015         1.00e+000
  27         8.75078e-002     2.40547e-006       1.50e-015         1.00e+000
  28         8.75074e-002     1.20273e-006       1.64e-015         1.00e+000
  29         8.75073e-002     6.01367e-007       1.60e-015         1.00e+000
  30         8.75072e-002     3.00683e-007       2.39e-016         1.00e+000
  31         8.75071e-002     1.50342e-007       1.63e-015         1.00e+000
  32         8.75071e-002     7.51708e-008       1.59e-015         1.00e+000
  33         8.75071e-002     3.75854e-008       1.64e-015         1.00e+000
  34         8.75071e-002     1.87927e-008       1.14e-016         1.00e+000
  35         8.75071e-002     9.39635e-009       1.75e-015         1.00e+000
x =
   68.6565
  -70.1784
  -70.1784
  -70.1784
```

```
   -70.1784
   -70.1784
status =
Solved
ans =
   6.5635e+029
```

$$P = \begin{bmatrix} 0,7 & 0,15 & 0,05 & 0,05 & 0,05 \\ 0,05 & 0,7 & 0,15 & 0,05 & 0,05 \\ 0,05 & 0,05 & 0,7 & 0,15 & 0,05 \\ 0,05 & 0,05 & 0,05 & 0,7 & 0,15 \\ 0,15 & 0,05 & 0,05 & 0,05 & 0,7 \end{bmatrix}$$

```
clear
clc
P=[0.7 0.15 0.05 0.05 0.05; 0.05 0.7 0.15 0.05 0.05 ;0.05 0.05 0.7 0.15 0.05;0.05 0.05 0.05 0.7
0.15;0.15 0.05 0.05 0.05 0.7]
[N,M] = size(P);
S=1
si=[ones(N,1)];
% variabel GP
gpvar z(M) w
% fungsi objektif adalah channel capacity
obj = w^S*sum(z)
% kendala
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <= w^si(k)*(prod( z.^(P(k,:)')) )
end
A0=[ones(N,1) eye(N)];
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1 -1*(P(k,:))]
%A=(An)'
bi(k,:)=log([exp(-entropy(P(k,:)) )])
%szs= (szs1)'
end
Ain=[-1 zeros(1,N)]
bin=log([1])
A=[A0;Ai;Ain]
b=[b0;bi;bin]
szs=[size(A0,1);ones(N+1,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
```

hasil program

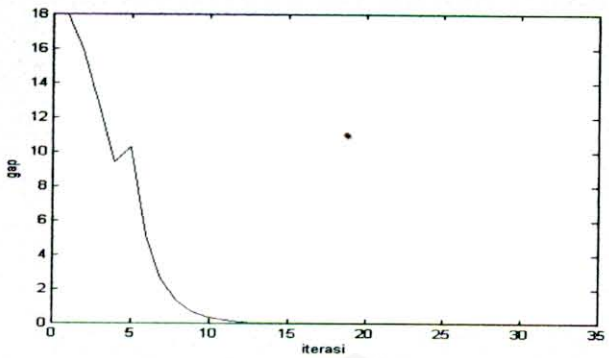
```
objektif is a posynomial with 5 monomial terms
 objektif   =   1*w^(1)*z_1^(1)   +   1*w^(1)*z_2^(1)   +   1*w^(1)*z_3^(1)   +   1*w^(1)*z_4^(1)   +
1*w^(1)*z_5^(1)
 kendala              is              a              constraint              0.24195              <=
1*w^(1)*z_1^(0.7)*z_2^(0.15)*z_3^(0.05)*z_4^(0.05)*z_5^(0.05)
kendala is a set of 5 constraints (index to get individual constraints).
Iteration     primal obj.          gap          dual residual     previous step.
   1         1.00000e+000     1.80000e+001      3.84e+000              Inf
   2        -6.69501e+001     1.68592e+001      3.78e+000         1.55e-002
Iteration     primal obj.          gap          dual residual     previous step.
   1         6.95783e+001     1.80000e+001      1.01e+000              Inf
   2         2.99830e+001     1.58668e+001      9.58e-001         4.68e-002
   3         3.92644e+000     1.25826e+001      8.53e-001         1.10e-001
   4         9.80733e-001     9.60869e+000      4.26e-001         5.00e-001
   5         3.83768e+000     1.00610e+001      3.12e-015         1.00e+000
   6         1.58781e+000     5.03052e+000      2.62e-015         1.00e+000
   7         8.89084e-001     2.51526e+000      5.21e-016         1.00e+000
   8         5.39755e-001     1.25763e+000      2.68e-015         1.00e+000
   9         3.65085e-001     6.28815e-001      4.74e-016         1.00e+000
  10         2.77749e-001     3.14407e-001      1.81e-015         1.00e+000
  11         2.34081e-001     1.57204e-001      4.90e-016         1.00e+000
  12         2.12248e-001     7.86019e-002      1.12e-015         1.00e+000
  13         2.01331e-001     3.93009e-002      1.78e-015         1.00e+000
  14         1.95872e-001     1.96505e-002      1.62e-015         1.00e+000
  15         1.93143e-001     9.82523e-003      1.30e-015         1.00e+000
  16         1.91778e-001     4.91262e-003      2.08e-015         1.00e+000
```

```
   17          1.91096e-001      2.45631e-003      2.20e-015          1.00e+000
   18          1.90755e-001      1.22815e-003      1.36e-015          1.00e+000
   19          1.90584e-001      6.14077e-004      9.02e-016          1.00e+000
   20          1.90499e-001      3.07039e-004      2.29e-015          1.00e+000
   21          1.90456e-001      1.53519e-004      2.04e-015          1.00e+000
   22          1.90435e-001      7.67596e-005      1.50e-015          1.00e+000
   23          1.90424e-001      3.83798e-005      1.69e-015          1.00e+000
   24          1.90419e-001      1.91899e-005      1.17e-015          1.00e+000
   25          1.90416e-001      9.59495e-006      1.37e-015          1.00e+000
   26          1.90415e-001      4.79748e-006      1.35e-015          1.00e+000
   27          1.90414e-001      2.39874e-006      6.57e-016          1.00e+000
   28          1.90414e-001      1.19937e-006      1.59e-015          1.00e+000
   29          1.90414e-001      5.99685e-007      2.05e-015          1.00e+000
   30          1.90414e-001      2.99842e-007      2.27e-015          1.00e+000
   31          1.90414e-001      1.49921e-007      1.95e-015          1.00e+000
   32          1.90414e-001      7.49606e-008      2.41e-015          1.00e+000
   33          1.90414e-001      3.74803e-008      1.41e-015          1.00e+000
   34          1.90414e-001      1.87401e-008      1.20e-015          1.00e+000
   35          1.90414e-001      9.37008e-009      2.25e-015          1.00e+000
x =
    68.7024
   -70.1214
   -70.1214
   -70.1214
   -70.1214
   -70.1214
status =
Solved
ans =
   6.8720e+029
```

$$P = \begin{bmatrix} 0{,}25 & 0{,}25 & 0{,}2 & 0{,}15 & 0{,}15 \\ 0{,}15 & 0{,}25 & 0{,}25 & 0{,}2 & 0{,}15 \\ 0{,}15 & 0{,}15 & 0{,}25 & 0{,}25 & 0{,}2 \\ 0{,}2 & 0{,}15 & 0{,}15 & 0{,}25 & 0{,}25 \\ 0{,}25 & 0{,}2 & 0{,}15 & 0{,}15 & 0{,}25 \end{bmatrix}$$

```
>>clear
>>clc
>>P=[0.25 0.25 0.2 0.15 0.15;0.15 0.25 0.25 0.2 0.15;0.15 0.15 0.25 0.25 0.2 ;0.2 0.15 0.15
0.25 0.25;0.25 0.2 0.15 0.15 0.25]
S=1
[N,M] = size(P);
si=[ones(N,1)];
% variabel GP
gpvar z(M) w
% fungsi objektif adalah channel capacity
obj = w^S*sum(z)
% kendala
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <= w^si(k)*(prod( z.^(P(k,:)')) )
end
A0=[ones(N,1) eye(N)];
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1 -1*(P(k,:))]
%A=(An)'
bi(k,:)=log([exp(-entropy(P(k,:)) )])
%szs= (szs1)'
end
Ain=[-1 zeros(1,N)]
bin=log([1])
A=[A0;Ai;Ain]
b=[b0;bi;bin]
szs=[size(A0,1);ones(N+1,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
hasil program
objektif is a posynomial with 5 monomial terms
```

```
 objektif   =   1*w^(1)*z_1^(1)   +   1*w^(1)*z_2^(1)   +   1*w^(1)*z_3^(1)   +   1*w^(1)*z_4^(1)   +
1*w^(1)*z_5^(1)
kendala is a set of 5 constraints (index to get individual constraints).
Iteration     primal obj.          gap        dual residual     previous step.
    1         1.00000e+000     1.80000e+001      3.05e+000             Inf
    2        -6.69548e+001     1.68590e+001      3.00e+000         1.55e-002
Iteration     primal obj.          gap        dual residual     previous step.
    1         6.95896e+001     1.80000e+001      1.01e+000             Inf
    2         2.97991e+001     1.58959e+001      9.60e-001         4.60e-002
    3         3.27899e+000     1.26283e+001      8.57e-001         1.08e-001
    4        -2.54661e-001     9.39477e+000      4.28e-001         5.00e-001
    5         3.22479e+000     1.03115e+001      6.20e-015         1.00e+000
    6         7.56270e-001     5.15575e+000      2.86e-015         1.00e+000
    7         4.00666e-002     2.57787e+000      3.65e-016         1.00e+000
    8        -3.17957e-001     1.28894e+000      4.10e-016         1.00e+000
    9        -4.96976e-001     6.44469e-001      1.97e-015         1.00e+000
   10        -5.86486e-001     3.22234e-001      9.62e-016         1.00e+000
   11        -6.31240e-001     1.61117e-001      1.84e-015         1.00e+000
   12        -6.53618e-001     8.05586e-002      1.69e-015         1.00e+000
   13        -6.64806e-001     4.02793e-002      1.61e-015         1.00e+000
   14        -6.70401e-001     2.01396e-002      1.48e-015         1.00e+000
   15        -6.73198e-001     1.00698e-002      1.90e-015         1.00e+000
   16        -6.74597e-001     5.03491e-003      1.91e-015         1.00e+000
   17        -6.75296e-001     2.51746e-003      1.61e-015         1.00e+000
   18        -6.75645e-001     1.25873e-003      1.43e-015         1.00e+000
   19        -6.75820e-001     6.29364e-004      1.62e-015         1.00e+000
   20        -6.75908e-001     3.14682e-004      8.25e-016         1.00e+000
   21        -6.75951e-001     1.57341e-004      1.06e-015         1.00e+000
   22        -6.75973e-001     7.86705e-005      1.10e-015         1.00e+000
   23        -6.75984e-001     3.93352e-005      1.38e-015         1.00e+000
   24        -6.75990e-001     1.96676e-005      6.93e-017         1.00e+000
   25        -6.75992e-001     9.83381e-006      1.61e-015         1.00e+000
   26        -6.75994e-001     4.91691e-006      2.32e-016         1.00e+000
   27        -6.75994e-001     2.45845e-006      1.53e-015         1.00e+000
   28        -6.75995e-001     1.22923e-006      9.30e-017         1.00e+000
   29        -6.75995e-001     6.14613e-007      1.53e-015         1.00e+000
   30        -6.75995e-001     3.07307e-007      1.62e-015         1.00e+000
   31        -6.75995e-001     1.53653e-007      1.61e-015         1.00e+000
   32        -6.75995e-001     7.68267e-008      6.59e-017         1.00e+000
   33        -6.75995e-001     3.84133e-008      1.53e-016         1.00e+000
   34        -6.75995e-001     1.92066e-008      1.54e-015         1.00e+000
   35        -6.75995e-001     9.60332e-009      1.32e-016         1.00e+000
Solved
x =
   68.3162
  -70.6017
  -70.6017
  -70.6017
  -70.6017
  -70.6017
ans =
  4.6706e+029
Plot gap 3.4
>>iterasi =1: 35
>>Gap=[1.80000e+001          1.58959e+001          1.26283e+001          9.39477e+000
1.03115e+001          5.15575e+000          2.57787e+000          1.28894e+000
6.44469e-001          3.22234e-001          1.61117e-001          8.05586e-002
4.02793e-002          2.01396e-002          1.00698e-002          5.03491e-003
2.51746e-003          1.25873e-003          6.29364e-004          3.14682e-004
1.57341e-004          7.86705e-005          3.93352e-005          1.96676e-005
9.83381e-006          4.91691e-006          2.45845e-006          1.22923e-006
6.14613e-007          3.07307e-007          1.53653e-007          7.68267e-008
3.84133e-008      1.92066e-008      9.60332e-009]
 >>plot(iterasi,Gap)
>>xlabel('iterasi');ylabel('gap');
```



6. $P = \begin{bmatrix} 0{,}02 & 0{,}02 & 0{,}02 & \cdots & 0{,}02 \\ 0{,}02 & 0{,}02 & 0{,}02 & \cdots & 0{,}02 \\ 0{,}02 & 0{,}02 & 0{,}02 & \cdots & 0{,}02 \cdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0{,}02 & 0{,}02 & 0{,}02 & \cdots & 0{,}02 \end{bmatrix}$, P_{50x50}

```
A=1/50;
P=A*ones(50);
S=1
[N,M] = size(P);
si=[ones(N,1)];
% variabel GP
gpvar z(M) w
% fungsi objektif adalah channel capacity
obj = w^S*sum(z)
% kendala
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <= w^si(k)*(prod( z.^(P(k,:)')) )
end
A0=[ones(N,1) eye(N)];
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1 -1*(P(k,:))]
%A=(An)'
bi(k,:)=log([exp(-entropy(P(k,:)) )])
%szs= (szs1)'
end
Ain=[-1 zeros(1,N)]
bin=log([1])
A=[A0;Ai;Ain]
b=[b0;bi;bin]
szs=[size(A0,1);ones(N+1,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
```

hasil program

```
objektif is a posynomial with 50 monomial terms
 objektif = 1*w^(1)*z_1^(1) + 1*w^(1)*z_2^(1) + 1*w^(1)*z_3^(1) + … + 1*w^(1)*z_50^(1)
 kendala            is            a          constraint          0.0035392          <=
1*w^(1)*z_1^(0.02)*z_2^(0.02)*z_3^(0.02)*z_4^(0.02)*z_5^(0.02)*z_6^(0.02)*z_7^(0.02)*z_8^(0.02)
*z…*z_49^(0.02)*z_50^(0.02)
kendala is a set of 50 constraints (index to get individual constraints).
Iteration     primal obj.          gap          dual residual     previous step.
   1         1.00000e+000     1.53000e+002       1.17e+001            Inf
   2        -1.18535e+002     1.14921e+002       8.80e+000         2.48e-001
Iteration     primal obj.          gap          dual residual     previous step.
   1         1.24269e+002     1.53000e+002       6.06e-001            Inf
   2         1.51979e+001     8.96240e+001       3.35e-001         4.46e-001
   3         2.79136e+001     8.07097e+001       2.79e-014         1.00e+000
   4         1.15596e+001     4.03548e+001       8.74e-014         1.00e+000
   5         4.86908e+000     2.01774e+001       1.71e-014         1.00e+000
   6         1.56529e+000     1.00887e+001       5.57e-015         1.00e+000
   7        -8.33478e-002     5.04436e+000       6.02e-015         1.00e+000
```

```
   8        -9.07590e-001     2.52218e+000     2.68e-015        1.00e+000
   9        -1.31971e+000     1.26109e+000     2.55e-015        1.00e+000
  10        -1.52577e+000     6.30544e-001     4.06e-015        1.00e+000
  11        -1.62880e+000     3.15272e-001     3.01e-016        1.00e+000
  12        -1.68032e+000     1.57636e-001     1.13e-015        1.00e+000
  13        -1.70607e+000     7.88181e-002     4.32e-015        1.00e+000
  14        -1.71895e+000     3.94090e-002     3.65e-015        1.00e+000
  15        -1.72539e+000     1.97045e-002     5.94e-016        1.00e+000
  16        -1.72861e+000     9.85226e-003     4.36e-015        1.00e+000
  17        -1.73022e+000     4.92613e-003     4.88e-016        1.00e+000
  18        -1.73103e+000     2.46306e-003     1.38e-015        1.00e+000
  19        -1.73143e+000     1.23153e-003     1.61e-015        1.00e+000
  20        -1.73163e+000     6.15766e-004     1.02e-015        1.00e+000
  21        -1.73173e+000     3.07883e-004     5.68e-017        1.00e+000
  22        -1.73178e+000     1.53942e-004     4.97e-016        1.00e+000
  23        -1.73181e+000     7.69708e-005     2.26e-015        1.00e+000
  24        -1.73182e+000     3.84854e-005     1.61e-015        1.00e+000
  25        -1.73183e+000     1.92427e-005     2.76e-016        1.00e+000
  26        -1.73183e+000     9.62134e-006     6.46e-016        1.00e+000
  27        -1.73183e+000     4.80389e-006     3.85e-010        1.00e+000
  28        -1.73183e+000     2.40149e-006     2.15e-010        1.00e+000
  29        -1.73183e+000     1.20075e-006     3.25e-011        1.00e+000
  30        -1.73183e+000     6.00377e-007     1.27e-011        1.00e+000
  31        -1.73183e+000     3.00194e-007     2.55e-010        1.00e+000
  32        -1.73183e+000     1.50097e-007     2.43e-011        1.00e+000
  33        -1.73183e+000     7.50485e-008     3.79e-012        1.00e+000
  34        -1.73183e+000     3.75243e-008     4.99e-012        1.00e+000
  35        -1.73183e+000     1.87621e-008     7.60e-012        1.00e+000
  36        -1.73183e+000     9.38107e-009     2.03e-012        1.00e+000
Solved
x =
   16.8814
  -22.5252
ans =
  2.1453e+007
```

.7. $P = \begin{bmatrix} 0{,}01 & 0{,}01 & 0{,}01 & \cdots & 0{,}01 \\ 0{,}01 & 0{,}01 & 0{,}01 & \cdots & 0{,}01 \\ 0{,}01 & 0{,}01 & 0{,}01 & \cdots & 0{,}01 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0{,}01 & 0{,}01 & 0{,}01 & \cdots & 0{,}01 \end{bmatrix}$, P100x100

```
A=1/100;
P=A*ones(100);
S=1
[N,M] = size(P);
si=[ones(N,1)];
% variabel GP
gpvar z(M) w
% fungsi objektif adalah channel capacity
obj = w^S*sum(z)
% kendala
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <= w^si(k)*(prod( z.^(P(k,:)')) )
end
A0=[ones(N,1) eye(N)];
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1 -1*(P(k,:))]
%A=(An)'
bi(k,:)=log([exp(-entropy(P(k,:)) )])
%szs= (szs1)'
end
Ain=[-1 zeros(1,N)]
bin=log([1])
A=[A0;Ai;Ain]
b=[b0;bi;bin]
szs=[size(A0,1);ones(N+1,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
```

hasil program

Iteration	primal obj.	gap	dual residual	previous step.
1	1.00000e+000	3.03000e+002	1.98e+001	Inf
2	-8.99325e+001	2.07366e+002	1.01e+001	4.93e-001

Iteration	primal obj.	gap	dual residual	previous step.
1	9.74178e+001	3.03000e+002	1.31e-002	Inf
2	4.93357e+001	1.52073e+002	1.31e-013	1.00e+000
3	2.32771e+001	7.60365e+001	1.39e-013	1.00e+000
4	1.05716e+001	3.80183e+001	5.40e-014	1.00e+000
5	4.24074e+000	1.90091e+001	6.56e-015	1.00e+000
6	1.09831e+000	9.50456e+000	1.08e-014	1.00e+000
7	-4.70271e-001	4.75228e+000	5.95e-015	1.00e+000
8	-1.25448e+000	2.37614e+000	5.43e-015	1.00e+000
9	-1.64658e+000	1.18807e+000	3.96e-015	1.00e+000
10	-1.84263e+000	5.94035e-001	3.39e-015	1.00e+000
11	-1.94066e+000	2.97018e-001	6.11e-016	1.00e+000
12	-1.98967e+000	1.48509e-001	6.82e-015	1.00e+000
13	-2.01418e+000	7.42544e-002	9.39e-016	1.00e+000
14	-2.02643e+000	3.71272e-002	6.51e-016	1.00e+000
15	-2.03256e+000	1.85636e-002	3.86e-015	1.00e+000
16	-2.03562e+000	9.28180e-003	1.05e-015	1.00e+000
17	-2.03715e+000	4.64090e-003	1.14e-015	1.00e+000
18	-2.03792e+000	2.32045e-003	1.72e-015	1.00e+000
19	-2.03830e+000	1.16023e-003	6.87e-015	1.00e+000
20	-2.03849e+000	5.80113e-004	8.83e-016	1.00e+000
21	-2.03859e+000	2.90056e-004	1.47e-015	1.00e+000
22	-2.03864e+000	1.45028e-004	2.98e-015	1.00e+000
23	-2.03866e+000	7.25141e-005	3.40e-015	1.00e+000
24	-2.03867e+000	3.62570e-005	7.56e-015	1.00e+000
25	-2.03868e+000	1.81285e-005	2.04e-015	1.00e+000
26	-2.03868e+000	9.06426e-006	3.38e-015	1.00e+000
27	-2.03868e+000	4.53213e-006	2.60e-015	1.00e+000
28	-2.03868e+000	2.26606e-006	1.73e-015	1.00e+000
29	-2.03868e+000	1.13303e-006	5.24e-016	1.00e+000
30	-2.03868e+000	5.66516e-007	4.37e-015	1.00e+000
31	-2.03868e+000	2.83258e-007	1.07e-014	1.00e+000
32	-2.03868e+000	1.41629e-007	4.26e-015	1.00e+000
33	-2.03868e+000	7.08145e-008	2.58e-015	1.00e+000
34	-2.03868e+000	3.54073e-008	5.80e-015	1.00e+000
35	-2.03868e+000	1.77036e-008	2.34e-014	1.00e+000
36	-2.03868e+000	8.85181e-009	3.70e-015	1.00e+000

```
ans =
  2.0715e+006
```

$$P = \begin{bmatrix} 0{,}005 & 0{,}005 & 0{,}005 & \cdots & 0{,}005 \\ 0{,}005 & 0{,}005 & 0{,}005 & \cdots & 0{,}005 \\ 0{,}005 & 0{,}005 & 0{,}005 & \cdots & 0{,}005 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0{,}005 & 0{,}005 & 0{,}005 & \cdots & 0{,}005 \end{bmatrix}, \; P_{200x200}$$

```
A=1/200;
P=A*ones(200);
S=1
[N,M] = size(P);
si=[ones(N,1)];
% variabel GP
gpvar z(M) w
% fungsi objektif adalah channel capacity
obj = w^S*sum(z)
% kendala
kendala = gpconstraint;
for k = 1:N
kendala(k) = exp(-entropy(P(k,:)) ) <= w^si(k)*(prod( z.^(P(k,:)')) )
end
A0=[ones(N,1) eye(N)];
b0=log(ones(N,1));
for k = 1:N
Ai(k,:)=[-1 -1*(P(k,:))]
%A=(An)'
bi(k,:)=log([exp(-entropy(P(k,:)) )])
%szs= (szs1)'
```

```
end
Ain=[-1 zeros(1,N)]
bin=log([1])
A=[A0;Ai;Ain]
b=[b0;bi;bin]
szs=[size(A0,1);ones(N+1,1)]
 [x,status]=gpcvx(A,b,szs);
z=exp(x);
sum(z)
```

hasil program

Iteration	primal obj.	gap	dual residual	previous step.
1	1.00000e+000	6.03000e+002	3.46e+001	Inf
2	-2.90580e+001	2.34805e+002	1.12e+000	9.68e-001
Iteration	primal obj.	gap	dual residual	previous step.
1	3.95299e+001	6.03000e+002	3.82e+000	Inf
2	5.17029e+001	2.55044e+002	3.18e-013	1.00e+000
3	4.02092e+001	1.27522e+002	1.54e-013	1.00e+000
4	1.89313e+001	6.37609e+001	1.35e-013	1.00e+000
5	8.27061e+000	3.18805e+001	1.60e-013	1.00e+000
6	2.94613e+000	1.59402e+001	5.77e-014	1.00e+000
7	2.98087e-001	7.97012e+000	2.25e-014	1.00e+000
8	-1.02379e+000	3.98506e+000	2.56e-015	1.00e+000
9	-1.68466e+000	1.99253e+000	4.64e-015	1.00e+000
10	-2.01510e+000	9.96265e-001	1.12e-014	1.00e+000
11	-2.18032e+000	4.98132e-001	4.78e-015	1.00e+000
12	-2.26293e+000	2.49066e-001	1.19e-014	1.00e+000
13	-2.30423e+000	1.24533e-001	9.81e-015	1.00e+000
14	-2.32488e+000	6.22665e-002	7.62e-015	1.00e+000
15	-2.33521e+000	3.11333e-002	9.66e-015	1.00e+000
16	-2.34037e+000	1.55666e-002	2.19e-015	1.00e+000
17	-2.34295e+000	7.78332e-003	1.94e-015	1.00e+000
18	-2.34424e+000	3.89166e-003	1.60e-015	1.00e+000
19	-2.34489e+000	1.94583e-003	1.19e-014	1.00e+000
20	-2.34521e+000	9.72915e-004	1.02e-015	1.00e+000
21	-2.34537e+000	4.86457e-004	2.01e-015	1.00e+000
22	-2.34545e+000	2.43229e-004	6.99e-015	1.00e+000
23	-2.34549e+000	1.21614e-004	5.94e-016	1.00e+000
24	-2.34551e+000	6.08072e-005	1.03e-014	1.00e+000
25	-2.34552e+000	3.04036e-005	9.92e-015	1.00e+000
26	-2.34553e+000	1.52018e-005	9.93e-015	1.00e+000
27	-2.34553e+000	7.60090e-006	1.12e-014	1.00e+000
28	-2.34553e+000	3.80045e-006	2.61e-014	1.00e+000
29	-2.34553e+000	1.90022e-006	5.58e-015	1.00e+000
30	-2.34553e+000	9.50112e-007	5.42e-016	1.00e+000
31	-2.34553e+000	4.75056e-007	1.10e-014	1.00e+000
32	-2.34553e+000	2.37528e-007	7.37e-015	1.00e+000
33	-2.34553e+000	1.18764e-007	2.05e-015	1.00e+000
34	-2.34553e+000	5.93820e-008	8.44e-015	1.00e+000
35	-2.34553e+000	2.96910e-008	6.52e-015	1.00e+000
36	-2.34553e+000	1.48455e-008	1.68e-015	1.00e+000
37	-2.34553e+000	7.42271e-009	2.75e-014	1.00e+000

```
Solved
ans =
  9.9558e+003
```

LAMPIRAN B

Bentuk M-File *ggplab* dan *gpcvx*

1. M-File *ggplab*

```
M-File gpvar
% GPVARS Displays all the GP variables defined in the workspace.
%%  GPVARS will list all the GP variables present in the workspace,
%   together with their size, bytes size, and class.
%gpvars__names = []; gpvars__whos = whos;
for gpvars__k = 1:length(gpvars__whos)
  if strcmp(gpvars__whos(gpvars__k).class, 'gpvar')
    gpvars__names = [gpvars__names gpvars__whos(gpvars__k).name ' '];
  end
end
disp(' ');
disp('Available GP variables are:')
disp(' ');
if isempty(gpvars__names)
  disp('  None.');
else
  eval(['whos ' gpvars__names])
end
disp(' ');
clear gpvars__names gpvars__whos gpvars__k;
M-File gp constraint
function constr = gpconstraint(varargin)
% GPCONSTRAINT is a GP constraint class constructor.
%%  Inequality operators between GP objects, such as positive
%   scalars, GP variables, monomials, posynomials, and generalized
%   posynomials (when valid) return GPCONSTRAINT objects.
%   These GPCONSTRAINT objects can be assigned to variables, for example,
%     constr1 = x*y <= z;
%     constr2 = x^5 == 1;
%   are two valid GP constraints.
%%  A set of constraint is represented as an array of GP constraints, i.e.,
%     constr_set = [constr1 constr2];
%%  You can also define GP constraints using:
%     constr = gpconstraint('constraint string')
%if nargin == 0 % create an empty constraint
  constr.lhs    = [];
  constr.type   = [];
  constr.rhs    = [];
  constr.gpvars = {};
  constr = class(constr,'gpconstraint');
  return;
end

if nargin == 1 % one argument
  % invoke copy constructor if the new object is gpconstraint
  if isa(varargin{1},'gpconstraint')
    constr = varargin{1};
    return;
  end
end
if nargin == 3
  % standard argument list
  constr.lhs      = varargin{1};
  constr.type     = varargin{2};
  constr.rhs      = monomial(varargin{3});
  if( isnumeric(constr.lhs) )
    constr.lhs    = monomial(constr.lhs);
    constr.gpvars = constr.rhs.gpvars;
  elseif( isa(constr.lhs,'gpvar') )
    constr.lhs    = monomial(constr.lhs);
    constr.gpvars = union(constr.lhs.gpvars, constr.rhs.gpvars );
  else
    constr.gpvars = union(constr.lhs.gpvars, constr.rhs.gpvars);
  end
  constr = class(constr,'gpconstraint');
  return;
end
M-File gpconstrain
```

```
function constr = gpconstraint(varargin)
% GPCONSTRAINT is a GP constraint class constructor.
%%  Inequality operators between GP objects, such as positive
%   scalars, GP variables, monomials, posynomials, and generalized
%   posynomials (when valid) return GPCONSTRAINT objects.
%   These GPCONSTRAINT objects can be assigned to variables, for example,
%     constr1 = x*y <= z;
%     constr2 = x^5 == 1;
%   are two valid GP constraints.
%%  A set of constraint is represented as an array of GP constraints, i.e.,
%     constr_set = [constr1 constr2];
%%  You can also define GP constraints using:
%     constr = gpconstraint('constraint string')
%if nargin == 0 % create an empty constraint
  constr.lhs    = [];
  constr.type   = [];
  constr.rhs    = [];
  constr.gpvars = {};
  constr = class(constr,'gpconstraint');
  return;
end
if nargin == 1 % one argument
  % invoke copy constructor if the new object is gpconstraint
  if isa(varargin{1},'gpconstraint')
    constr = varargin{1};
    return;
  end
end
if nargin == 3
  % standard argument list
  constr.lhs      = varargin{1};
  constr.type     = varargin{2};
  constr.rhs      = monomial(varargin{3});
  if( isnumeric(constr.lhs) )
    constr.lhs    = monomial(constr.lhs);
    constr.gpvars = constr.rhs.gpvars;
  elseif( isa(constr.lhs,'gpvar') )
    constr.lhs    = monomial(constr.lhs);
    constr.gpvars = union(constr.lhs.gpvars, constr.rhs.gpvars );
  else
    constr.gpvars = union(constr.lhs.gpvars, constr.rhs.gpvars);
  end
  constr = class(constr,'gpconstraint');
  return;
end
M-File gpsolve
function [obj_value, solution, status] = gpsolve(varargin)
% GPSOLVE Solves a geometric programming optimization problem.
%%  GPSOLVE calls the internal primal-dual interior point solver
%   in order to solve the specified GP problem.
%   GPSOLVE calling sequence is:
%%  [obj_value, solution, status] = gpsolve(obj, constr_array, flag)
%%  where inputs are:
%   obj          - objective function for the GP problem
%   constr_array - array of problem constraints
%   flag         - 'min' or 'max' (optional, default is 'min')
%%  and outputs are:
%   obj_value - the optimal objective value (a number)
%   solution  - a cell array of GP variable names and their optimal values
%   status    - the problem status flag
%%  The status can be 'Solved' (if the optimization was successful),
%   'Infeasible' (if the problem was determined to be infeasible), and
%   'Failed' (if the optimization was not successful).
%%  The inputs can also be empty arrays. If the objective is an empty
%   array or a constant, then GPSOLVE solves a feasibility problem.
%   If the constraint array is empty, then we have an unconstrained GP.
%%  Internally GPSOLVE creates a GP problem object (gpproblem) and
%   calls its solve method.
%if nargin == 2
  obj    = varargin{1};
  constr = varargin{2};
  flag   = 'min';
elseif nargin == 3
  obj    = varargin{1};
  constr = varargin{2};
  flag   = varargin{3};
```

```
else
  error('Wrong number of input arguments.')
end
gp_problem_obj = gpproblem(obj, constr, flag);
result_obj = solve(gp_problem_obj);
obj_value = result_obj.obj_value;
solution  = result_obj.solution;
status    = result_obj.status;
```

2. M-File gpcvx

```
function [x,status,lambda,nu,mu] = gpcvx(A,b,szs,varargin)
% [x,status,lambda,nu,mu] = gpcvx(A,b,szs,G,h,l,u,quiet)
% solves the geometric program in convex form
%  minimize     lse(y0)
%  subject to   lse(yi) <= 0,    i=1,...,m,
%               Ai*x+bi = yi,    i=0,...,m,
%               G*x+h = 0,
%               li <= xi <= ui, i=1,...,n
% where lse is defined as  lse(y) = log sum_i exp yi,
% and the dual problem,
%  maximize     b0'*nu0 + ... + bm'*num + h'*mu + lambdal'*l - lambdau'*u +
%                  entr(nu0) + lambdal*entr(nul/lambdal) +
%                  ,..., + lambdam*entr(num/lambdam)
%  subject to   nui >= 0,          i=0,...,m,
%               lambdai >= 0,      i=1,...,m,
%               1'*nu0 = 1
%               1'*nui = lambdai, i=1,...,m,
%               A0'*nu0 + ... + Am'*num + G'*mu + lambdau - lambdal = 0,
% where entr is defined as  entr(y) = -sum_i yi*log(yi).
%% Calling sequences:
%%  [x,status,lambda,nu,mu] = gpcvx(A,b,szs)
%  [x,status,lambda,nu,mu] = gpcvx(A,b,szs,G,h)
%  [x,status,lambda,nu,mu] = gpcvx(A,b,szs,G,h,l,u)
%  [x,status,lambda,nu,mu] = gpcvx(A,b,szs,G,h,l,u,quiet)
%% Examples:
%%  [x,status,lambda,nu,mu] = gpcvx(A,b,szs,G,h)
%  [x,status,lambda,nu,mu] = gpcvx(A,b,szs,[],[],[],[],quite)
%% Input arguments:
%% - A:          (sum_i n_i) x n matrix;  A   = [A0' A1' ... Am' ]'
% - b:          (sum_i n_i) vector;      b   = [b0' b1' ... bm' ]'
% - szs:        dimensions of Ai and bi; szs = [n0 n1 ... nm]'
%               where Ai is (ni x n) and bi is (ni x 1)
% - G:          p x n matrix
% - h:          p-vector
% - l:          n-vector; lower bound for x
% - u:          n-vector; upper bound for x
% - quiet:      boolean variable; suppress all the print messages if true
% Output arguments:
% - x:          n-vector; primal optimal point
% - nu:         (sum_i n_i) vector;  nu = [nu0' nu1' ... num']'
%               dual variables for constraints Ai*x + bi = yi
% - mu:         (sum_i l_i) vector;  mu = [mu0' mu1' ... mul']'
%               dual variables for constraints G*x + h = 0
% - lambda:     m-vector, dual variables for inequality constraints
% - status:     string; 'Infeasible', 'Solved', or 'Failed'
%
% gpcvx sets up phase 1 and phase 2 and calls the real sovler, gppd2.m.
%-------------------------------------------------------------------------
%       INITIALIZATION
%-------------------------------------------------------------------------
% PARAMETERS
LOWER_BOUND = -250;
UPPER_BOUND = +250;
% DIMENSIONS
N  = size(A,1);      % # of terms in the obj. and inequalities
n  = size(A,2);      % # of variables (x1,..,xn)
m  = length(szs)-1; % # of inequalities
n0 = szs(1);         % # of terms in the objective
% VARIABLE ARGUMENT HANDLING
defaults  = {[],[],LOWER_BOUND*ones(n,1),UPPER_BOUND*ones(n,1),false};
givenArgs = ~cellfun('isempty',varargin);
defaults(givenArgs) = varargin(givenArgs);
[G,h,l,u,quiet]  = deal(defaults{:});
% MATLAB LIMIT OF LOWER/UPPER BOUNDS
l(l<LOWER_BOUND) = LOWER_BOUND;
u(u>UPPER_BOUND) = UPPER_BOUND;
if (isempty(G))
```

```
    G = zeros(0,n);
    h = zeros(0,1);
end
p  = size(G,1);     % # of (terms in) the equality constraints
% E is a matrix s.t. [1'*y0  1'*y1  ... 1'*ym ]' = E*y
indsl = cumsum(szs);
indsf = indsl-szs+1;
lx    = zeros(N,1);
lx(indsf) = 1;
E = sparse(cumsum(lx),[1:N],ones(N,1));
%-------------------------------------------------------------------------
%                PHASE I
%-------------------------------------------------------------------------
% solves the feasibility problem
%%  minimize    s
%  subject to  lse(yi) <= s,      i=1,...,m,
%              Ai*x+bi = yi,      i=0,...,m,
%              G*x+h = 0,
%              li <= xi <= ui,    i=1,...,n
%% where lse is defined as  lse(y) = log sum_i exp yi,
%% For phase I
% 1) change objective function to s
% 2) change constraints from fi(x) <= 0 to fi(x) <= s
% 3) add bound constraints; li <= xi <= ui
%% Hence, we set up a new objective and constraints,
%    i.e., A,b,G,h and szs for Phase I optimization.
%% Change the size vector, szs.
%% Change A and b
%          s    xi
%   A1 = [ 1 | 0 0 0      b1 = [ 0      <- (a1) new objective
%          ---+------            -
%          -1 |
%          -1 |A(ineq)           b      <- (a2) new inequalities
%          -1 |
%          ---+------            -
%          -1 |-1 0 0            l1
%          -1 | 0-1 0            l2     <- (a4) l <= xi
%          -1 | 0 0-1            l3
%          ---+------            -
%          -1 | 1 0 0           -u1
%          -1 | 0 1 0           -u2     <- (a6) xi <= u
%          -1 | 0 0 1 ];        -u3 ];
% FORM SZS
szs1    = [1; szs(2:end); ones(2*n,1) ];
% FORM INITIAL X
if (p == 0)
    xinit = zeros(n,1);
else
    xinit = G'*((G*G')\h);
end
% FORM INITIAL S
%  sinit = max(fi,0) since fi <= si and 0 <= si
y = A*xinit+b;
[f,expyy] = lse(E,y);
finit = f(2:m+1);
linit = -xinit+l;
uinit = +xinit-u;
sinit = max([0; finit; linit; uinit]) + 1; % + 1 is for margin.
% FORM A AND B
A1 = [+1            , sparse(1,n);...
      -ones(N-n0,1) , A(n0+1:N,:);...
      -ones(n,1)    ,-speye(n)   ;...
      -ones(n,1)    ,+speye(n)   ];
b1 = [ 0; b((n0+1):N); l; -u ];
% FORM G AND H
G1 = [spalloc(size(G,1),1,0), G];
h1 = [h];
% CALL THE INTERNAL GP SOLVER
[x,status,lambda,nu,mu] = gppd2(A1,b1,szs1,[sinit;xinit],G1,h1,true,quiet);
% EXTRACT X FROM [S; X]
x0 = x(2:n+1);
y = A*x0+b;
[f,expyy] = lse(E,y);
flm       = f(2:m+1);
% FEASIBILITY CHECK OF PHASE I SOLUTION
if (status <= 0 || max([flm; -Inf]) >= 0)
```

```
    status = 'Infeasible';
    if (~quiet) disp(status); end
    return
end
clear A1 b1 G1 h1 x01 szs1;
%-------------------------------------------------------------------------
%               PHASE II
%-------------------------------------------------------------------------
% solves the geometric program in convex form
%%  minimize    lse(y0)
%  subject to  lse(yi) <= 0,   i=1,...,m,
%              Ai*x+bi = yi,   i=0,...,m,
%              G*x+h = 0,
%              li <= xi <= ui, i=1,...,n
%% where lse is defined as  lse(y) = log sum_i exp yi,
%% Change A and b to add the bound of x into the inequality constraints.
%
%   A2 = [  A         b2 = [ b
%         ------             ----
%         -1 0 0              l1
%          0-1 0              l2      <- li <= xi
%          0 0-1              l3
%         ------             ----
%          1 0 0             -u1     <- xi <= ui
%          0 1 0             -u2
%          0 0 1 ];          -u3 ];
szs2 = [ szs ; ones(2*n,1) ];
A2   = [ sparse(A); -speye(n); speye(n) ];
b2   = [ b; l; -u ];
% CALL THE INTERNAL GP SOLVER
[x,status,lambda,nu,mu] = gppd2(A2,b2,szs2,x0,G,h,false,quiet);
if (status <= 0)
    status = 'Failed';
    if (~quiet) disp(status); end
    return
else
    status = 'Solved';
    if (~quiet) disp(status); end
    return
end
```

M-File gppd pada gpcvx

```
function [x,status,la,nu,mu] = gppd2(A,b,szs,x0,G,h,phase1,quiet)
% [x,nu,mu,la,status] = gppd2(A,b,szs,x0,G,h,phase1,quiet)
% solves the geometric program in convex form with a starting point.
%  minimize    lse(y0)
%  subject to  lse(yi) <= 0,   i=1,...,m,
%              Ai*x+bi = yi,   i=0,...,m,
%              G*x+h = 0,
% where lse is defined as  lse(y) = log sum_i exp yi,
% and the dual problem,
%  maximize    b0'*nu0 + ... + bm'*num + h'*mu +
%                 entr(nu0) + la1*entr(nu1/la1) +
%                 ,..., + lam*entr(num/lam)
%  subject to  nui >= 0,          i=0,...,m,
%              lai >= 0,      i=1,...,m,
%              1'*nu0 = 1
%              1'*nui = lai, i=1,...,m,
%              A0'*nu0 + ... + Am'*num + G'*mu = 0,
% where entr is defined as  entr(y) = -sum_i yi*log(yi).
% x0 should satisfy the primal inequality constraints.
% Input arguments:
% - A:          (sum_i n_i) x n matrix; A = [A0' A1' ... Am' ]'
% - b:          (sum_i n_i) vector;   b = [b0' b1' ... bm' ]'
% - szs:        dimensions of Ai and bi; szs = [Nob n1 ... nm]'
%               where Ai is (ni x n) and bi is (ni x 1)
% - x0:         n-vector; MUST BE STRICTLY FEASIBLE FOR INEQUALITIES
% - G:          p x n matrix
% - h:          p-vector
% - phase1:     boolean variable; indicator for phase I and phase II
%               true -> Phase I, false -> Phase II
% - quiet:      boolean variable; suppress all the print messages if true
% Output arguments:
% - x:          n-vector; primal optimal point
% - nu:         (sum_i n_i) vector;  nu = [nu0' nu1' ... num']'
%               dual variables for constraints Ai*x + bi = yi
% - mu:         p vector; mu = [mu1 ... mup]'
```

```
%               dual variables for constraints G*x + h = 0
% - la:         m vector; la = [lambda1 ... lambdam]'
%               dual variables lambda; la_i = sum(nu_i)
% - status:     scalar;
%               2        Function converged to a solution x.
%               1        Phase I success; x(1:szs(1)) <= 0.
%              -1        Number of iterations exceeded MAXITERS.
%              -2        Starting point is not strictly feasible.
%              -3        Newton step calculation failure.
%--------------------------------------------------------------------
%                INITIALIZATION
%--------------------------------------------------------------------
% PARAMETERS
ALPHA   = 0.01;     % backtracking linesearch parameter (0,0.5]
BETA    = 0.5;      % backtracking linesearch parameter (0,1)
MU      = 2;        % IPM parameter: t update
MAXITER = 500;      % IPM parameter: max iteration of IPM
EPS     = 1e-8;     % IPM parameter: tolerance of surrogate duality gap
EPSfeas = 1e-8;     % IPM parameter: tolerance of feasibility
% DIMENSIONS
[N,n] = size(A); m = length(szs)-1; p = size(G,1); n0 = szs(1);
if (isempty(G)), G = zeros(0,n); h = zeros(0,1); end
warning off all;
% SPARSE ZERO MATRIX
Opxp = sparse(p,p);
% SUM MATRIX: E is a matrix s.t. [1'*y0  1'*y1  ... 1'*ym ]' = E*y
indsl = cumsum(szs);
indsf = indsl-szs+1;
lx    = zeros(N,1);
lx(indsf) = 1;
E = sparse(cumsum(lx),[1:N],ones(N,1));
x = x0;
% flm is a LHS vector of inequality constraints s.t [f1' ... fm']'
y = A*x+b;
[f,expyy] = lse(E,y);
flm       = f(2:m+1);
% CHECK THE INITIAL CONDITIONS
if (max(flm) >= 0)
   if (~quiet) disp(['x0 is not strictly feasible.']); end
   la = []; nu = []; mu = [];
   status = -2;
   return;
end;
% INITIAL DUAL POINT
la = -1./flm;   % positive value with duality gap 1.
nu = ones(N,1); % ANY value.
mu = ones(p,1); % ANY value.
step = Inf;
if (~quiet) disp(sprintf('\n%s %15s %11s %20s %18s \n',...
    'Iteration','primal obj.','gap','dual residual','previous step.')); end
%--------------------------------------------------------------------
%                MAIN LOOP
%--------------------------------------------------------------------
for iters = 1:MAXITER
    gap = -flm'*la;
    % UPDATE T
    % update t only when the current x is not to far from the cetural path.
    if (step > 0.5)
        t = m*MU/gap;
   end
    % CALCULATE RESIDUALS
    % gradfy = exp(y)./(E'*(E*exp(y)));
    gradfy   = expyy./(E'*(E*expyy));
    resDual  = A'*(gradfy.*(E'*[1;la])) + G'*mu;
    resPrim  = G*x + h;
    resCent  = [-la.*flm-1/t];
    residual = [resDual; resCent; resPrim];
    if (~quiet) disp(sprintf('%4d %20.5e %16.5e %14.2e %16.2e',...
        iters,f(1),-flm'*la,norm(resDual),step)); end
    % STOPPING CRITERION FOR PHASE I
    if ((phase1 == true) & max(x(1:szs(1))) < 0)
        nu = gradfy.*(E'*[1;la]);
        status = 1;
        return;
    end;
    % STOPPING CRITERION FOR PHASE I & II
```

```
    if ( (gap <= EPS) & ...
            (norm(resDual) <= EPSfeas) & (norm(resPrim) <= EPSfeas) )
        % this calculation of nu is correct only when reached to optimal
        nu = gradfy.*(E'*[1;la]);
        status = 2;
        return;
    end;
    % CALCULATE NEWTON STEP
    diagL1  = sparse(1:m+1,1:m+1,[0;-la./flm]);
    diagL2  = sparse(1:m+1,1:m+1,[1;la]);
    diagG1  = sparse(1:N,1:N,gradfy);
    diagG2  = sparse(1:N,1:N,gradfy.*(E'*[1;la]));
    EGA = E*diagG1*A;
    H1  = EGA'*(diagL1-diagL2)*EGA + A'*diagG2*A;
    dz  = -[ H1, G'   ; ...
             G , Opxp ]...
          \ ...
          [EGA'*[1;-1./(t*flm)]+G'*mu ; resPrim ];
    % PERTURB KKT WHEN (ALMOST) SINGULAR
    %  add small diagonal terms when the matrix is almost singular.
    perturb = EPS;
    while (any(isinf(dz)))
       dz = -([ H1, G'; G , Opxp ] + sparse(1:n+p,1:n+p,perturb))...
            \ ...
            [EGA'*[1;-1./(t*flm)]+G'*mu ; resPrim ];
        % increase the size of diagonal term if still singular
        perturb = perturb*10;
    end
    % ERROR CHECK FOR NEWTON STEP CALCULATION
    if (any(isnan(dz)))
        nu = gradfy.*(E'*[1;la]);
        status = -3;
        return;
    end
    dx  = dz(1:n); dy  = A*dx; dmu = dz(n+[1:p]');
    dla = -la./flm.*(E(2:m+1,:)*(gradfy.*dy))+resCent./flm;
    % BACKTRACKING LINESEARCH
    negIdx = (dla < 0);
    if (any(negIdx))
        step = min( 1, 0.99*min(-la(negIdx)./dla(negIdx)) );
    else
        step = 1;
    end
    while (1)
        newx    = x  + step*dx;
        newy    = y  + step*dy;
        newla   = la + step*dla;
        newmu   = mu + step*dmu;
        [newf,newexpyy] = lse(E,newy);
        newflm  = newf(2:end);
        % UPDATE RESIDUAL
        % newGradfy = exp(newy)./(E'*(E*exp(newy)));
        newGradfy = newexpyy./(E'*(E*newexpyy));
        newResDual  = A'*(newGradfy.*(E'*[1;newla])) + G'*newmu;
        newResPrim  = G*newx + h;
        newResCent  = [-newla.*newflm-1/t];
        newResidual = [newResDual; newResCent; newResPrim];
        if ( max(newflm) < 0 && ...
             norm(newResidual) <= (1-ALPHA*step)*norm(residual) )
            break;
        end
        step = BETA*step;
    end;
    % UPDATE PRIMAL AND DUAL VARIABLES
    x  = newx; mu = newmu; la = newla; y = A*x+b;
    [f,expyy] = lse(E,y); flm = f(2:m+1);
end
if (iters >= MAXITER)
    if (~quiet) disp(['Maxiters exceeded.']); end
    nu = gradfy.*(E'*[1;la]);
    status = -1;
    return;
end
```