MOKSLAS – LIETUVOS ATEITIS
SCIENCE – FUTURE OF LITHUANIA
ISSN 2029-2341 / eISSN 2029-2252
http://www.mla.vgtu.lt

ELEKTRONIKA IR ELEKTROTECHNIKA
ELECTRONICS AND ELECTRICAL ENGINEERING

2015 7(3): 280–284

Sistemų inžinerija, kompiuterių technologija T 120
System Engineering, Computer Technologies T 120

http://dx.doi.org/10.3846/mla.2015.782

APPLICATION OF SEMIDEFINITE PROGRAMMING TO TRUSS DESIGN OPTIMIZATION

Rasa GINIŪNAITĖ
University of Warwick, Coventry, UK
E-mail rasaginiu@gmail.com

Abstract. Semidefinite Programming (SDP) is a fairly recent way of solving optimization problems which are becoming more and more important in our fast moving world. It is a minimization of linear function over the intersection of the cone of positive semidefinite matrices with an affine space, i.e. non-linear but convex constraints. All linear problems and many engineering and combinatorial optimization problems can be expressed as SDP, so it is highly applicable. There are many packages that use different algorithms to solve SDP problems. They can be downloaded from internet and easily learnt how to use, two of these are SeDuMi and SDPT-3. In this paper truss structure optimization problem with the goal of minimizing the mass of the truss structure was solved. After doing some algebraic manipulation the problem was formulated suitably for semidefinite programming. SeDuMi and SDPT-3 packages were used to solve it. The choice of the initial solution had a great impact on the result using SeDuMi. The mass obtained using SDPT-3 was on average smaller than the one obtained using SeDuMi. Moreover, SDPT-3 worked more efficiently. However, the comparison of my approach and two versions of particle swarm optimization algorithm implied that semidefinite programming is in general more appropriate for solving such problems.

Keywords: Semidefinite Programming, truss design, SeDuMi, SDPT-3.

Introduction

Let's first introduce the notion of optimisation, which is well-known to most of the people as the procedure of making a system or design as effective or as functional as possible. Speaking more mathematically it involves finding the minimum/maximum of an objective function $f(x)$ subject to some constraint on variable $x \in S$. If there is no constraint for x, i.e. S is the universe, then it is called unconstrained optimisation. There are many different techniques for solving optimisation problems which have to be chosen regarding to whether the function is linear or nonlinear, discrete or continuous, the gradient is known or not, etc. Therefore it is crucial to define and understand the problem thoroughly in order to apply the most suitable method. In this paper truss design optimisation problem which is common in building bridges and cranes is considered. There are many parameters that have to be taken into account in optimising the structure (cross-sectional area and length of bars, compliance, etc.). In order to include all given constraints this problem will be regarded as semidefinite programming one. Two different packages will be used to solve it and then their results and efficiencies will be compared. Moreover, the results obtained using two versions of particle swarm optimisation algorithm will be briefly analysed (Tang *et al*. 2009; Yancang *et al.* 2013)

Semidefinite Programming

Semidefinite Programming (SDP) is a type of optimisation in which one minimises linear objective function subject to the constraint that an affine combination of symmetric matrices is positive definite (Vandenberghe, Boyd 1996). Affine combination of a finite set of vectors $v_1,...,v_n \in V$ (any vector space) is linear combination of the vectors $k_1v_1 + ... + k_nv_n$ such that $k_i \in V$ (division ring) subject to the condition $k_1 + \cdots + k_n = 1$. SDP problems arise from the well-known linear programming problems by replacing the vector of variables with a symmetric matrix and replacing the non-negativity constraints with positive semidefinite constraints. Under different names SDP has already been studied in 1940s, officially, it is believed that Bellman and Fan were the first who formulated semidefinite programming problem in 1963 (Todd 2001). However, there has been a significant increase of interest in this area in recent years. Let's consider some simple examples in order to understand the definition better.

Example 1.

Minimize $\underline{c}^T \underline{x}$,
subject to $F(\underline{x}) \geq 0$,

where $F(\underline{x}) = F_0 + \sum_{i=1}^{m} x_i F$ and $\underline{x} \in \Re^m$ is a variable, $\underline{c} \in \Re^m$ and symmetric matrices $F_i \in \Re^{n \times n}$ for $i = 0,...,m$ are problem data. Note that $F(\underline{x}) \geq 0$ means that $F(\underline{x})$ is positive semidefinite, i.e. $\forall \underline{z} \in \Re^n$, $\underline{z}^T F(\underline{x}) \underline{z} \geq 0$.

Example 2.

The more specific example could be to minimize the maximum eigenvalue (Todd 2001). The purpose of this problem might be to stabilize a differential equation. Let $M(\underline{z})$ be symmetric matrix which linearly depends on a vector $\underline{z}$. The vector $\underline{z}$ value which would minimize the maximum eigenvalue of $M(\underline{z})$ is of interest. Note that $\lambda_{\max}(M(\underline{z})) \leq \eta$ if and only if $\lambda_{\max}(M(\underline{z}) - \eta I) \leq 0$ or equivalently if and only if $\lambda_{\min}(\eta I - M(\underline{z})) \geq 0$. This holds if and only if $\eta I - M(\underline{z}) \geq 0$. So we can get SDP problem:

minimise η,
subject to $\eta I - M(\underline{z}) \geq 0$.

SeDuMi and SDPT-3

SeDuMi which stands for self-dual minimization is software for optimisation over symmetric cones (Sturm 1999). It can be used for solving optimisation problems with linear, quadratic and semidefinite constraints. SeDuMi package uses interior point method (using barrier function). SeDuMi gives solution to both primal and dual problems. The problem has to be of the form:

minimize $\underline{c}^T \underline{x}$,
subject to: $A\underline{x} = \underline{b}$ and $\underline{x} \geq 0$,
and it's dual
maximize $b^T \underline{x}$,
subject to $-A^T \underline{y} \geq -\underline{c}$.

Problem is solved using Matlab command `[x,y,info] = sedumi(A,b,c)`. Therefore if a problem with inequality constraints appears it is important to manipulate problem data and choose *b* and *c* vectors and their signs correctly. Now let's consider some examples.

Example 3.

Minimise $f(\underline{x}) = 3x_1 + 9x_2 + 3x_3$,
subject to: $2x_1 + 2x_2 + x_3 - x_4 = 1$,
$x_1 + x_4 - x_3 + x_5 = 1$,
$x_i \geq 0, i = 1,\ldots,5$.

Matlab code for solution of this problem using SeDuMi:

```
A = [3 2 1 -1 0; 1 4 -1 0 1];
b = [1 1]';
c = [2 9 3 0 0]';
[x,y] = sedumi(A,b,c);
```

The answer is `x = [0.3333 0 0 0 0.6667]'`.

The idea of using SeDuMi for SDP is similar to linear case. Considering the formulation of the problem based on Example 1, dual problem have to be used and since SeDuMi solves minimisation, not maximisation problem, minus sign will appear.

Example 4.

```
p = lenght(c);
bt = -c;
ct = vec(F0);
for i = 1:p, A(:,i) = -vec(Fi); end;
K.s = size(F0,1);
[x,y,info] = sedumi(A,bt,ct,K);
```

Note that `K.s` lists the dimensions of positive semidefinite constraints. And the desired solution will be `y`, since the problem described was dual.

SDPT-3 is used in a very similar way, everything is the same as in Example 4 apart from the last line, where following have to be written.

Example 5.

```
K.l = size(F0, 1);
[blk,Att,ctt,btt] = read_sedumi(A,bt,ct,K);
[obj,X,y,Z] = sqlp(blk,Att,ctt,btt);
```

Where `K.l` is the number of nonnegative components, and `blk` is a cell array which describes the block structure of the problem. Here `y` gives desired value.

Truss design optimization

A truss is a structure in $d = 2$, $d = 3$ dimensions, formed by *n* nodes and *m* bars joining these nodes. Specific example can be seen in Figure 1.

Trusses are widely used in constructing bridges, cranes and even Eiffel tower. Let's investigate the planar structure of 37 bars which is an approximation of a simply supported bridge (Tang *et al.* 2009). The initial configuration of the structure is shown in Figure 2. In the optimisation process, nodes (3, 5, 7, 9, 11, 13, 15, 17, 19) of the upper chord can be shifted vertically, while nodes (2, 4, 6, 8, 10, 12, 14, 16, 18) of the lower chord remain fixed. The

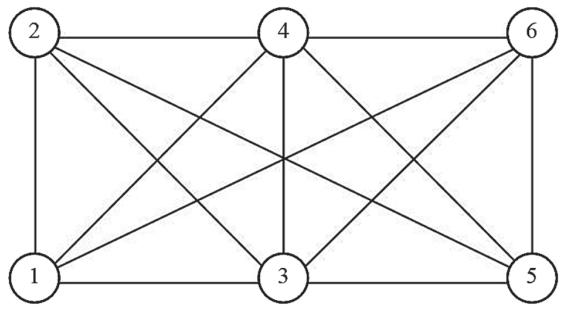

Fig. 1. Truss structure: d = 2, n = 6 nodes and m = 13 bars

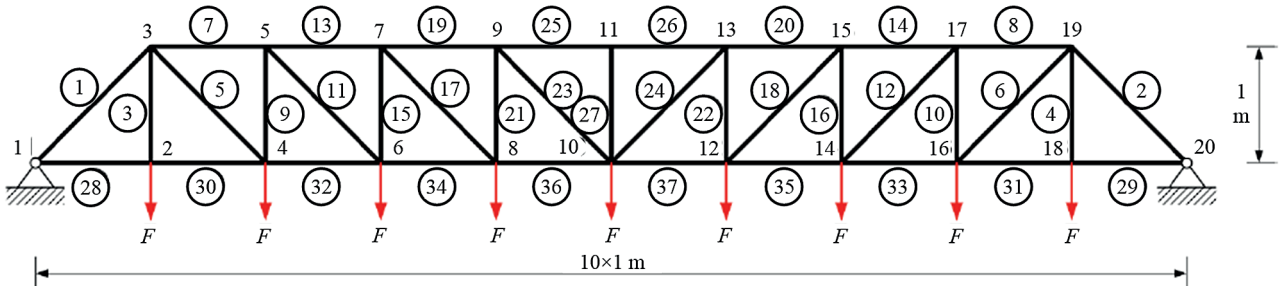

Fig. 2. The initial configuration of the structure

The loads of free nodes of the lower chord are $F = 10$ kN. The Young's modulus is $E = 210$ GPa, the material density is $\rho = 7800$ kg/m^3. Suppose that the topology of the structure is fixed. The nodal coordinates Y_j, j = 3, 5,…, 19 and the bar cross-sectional areas A_i, i = 1,…, 37 are the optimisation variables. The nodal coordinates and member areas are linked to maintain the structural symmetry. Of course cross-sectional area and nodal coordinates are bounded: 50 mm^2 ≤ A_i ≤ 1250 mm^2 and 250 mm^2 ≤ Y_i ≤ 3250 mm^2. The optimal shape of a truss structure implies that all the specified nodal displacements and bar stresses satisfy the prescribed constraints while the mass is minimal, where the mass of this structure is:

$$M = \rho \sum_{i=1}^{37} A_i L_i \,, \tag{1}$$

where L_i, $i = 1,...,37$ is the length of the ith bar. Let $\underline{u} \in \Re^{36}$ denote a vector $\underline{u} = \left[u_{2x}\ u_{2y} \ldots u_{19x}\ u_{19y}\right]$ which describes the displacement of each node. There is one more constraint of the vertical displacement of the node $j = 10$: $|u_{10y}| \le 10$ mm. So now this problem can be formulated fully:

minimize M,

subject to: $\underline{f}^T \underline{u} \le t$, (*)

$t \le a$,

$50\ \text{mm}^2 \le A_i \le 1250\ \text{mm}^2$,

$250\ \text{mm} \le Y_i \le 3250\ \text{mm}$,

$|u_{10y}| \le \text{mm}$.

Here $a \in \Re$ is a bound and $t \in \Re$ is another variable which will be crucial for getting semidefinite constraint. $\underline{f} \in \Re^{36}$ denotes vector $\underline{f} = \left[f_{2x}\ f_{2y} \ldots f_{19x}\ f_{19y}\right]$ and $f_{ix} = 0, \forall i = 2,\ldots,19$ and $f_{iy} = -10000$ for $i = 2,4,\ldots,18$ and $f_{iy} = 0$ for other i. The equilibrium equation of the mechanical problem is written as follows:

$$\sum_{i=1}^{m} A_i K_i \underline{u} - \underline{f} = 0 \,. \tag{2}$$

Where $K_i \in \Re^{18\times 18}$, $i = 1,\ldots,37$ are stiffness matrices of the bar elements:

$$K_i = \frac{E}{L_i} \underline{b}_i \underline{b}_i^T \,, \tag{3}$$

where $\underline{b}_i \in \Re^{18}$, $\underline{b}_i = \left[\cos(\alpha)\ \sin(\alpha) - \cos(\alpha) - \sin(\alpha)\right]$ at the location of the states of the nodes which are connected by bar i and α is the angle between bar i and the horizontal axis. The variable t is bounded, since the work done by external forces has to be limited, otherwise the solution would be trivial: the smallest cross-sectional area and the shortest length could be taken. It will have to be chosen with regards to bound on $\underline{u}$. Let vector $\underline{x} \in \Re^{37}$, $\underline{x} = \left[A_1 \ldots A_{37}\right] = \left[x_1 \ldots x_{37}\right]$ be optimization variable. To bound the work done by external forces, SDP has to be used, using equations (1) and (2) we obtain:

$$[A(\underline{x}, t)] \ge 0 \,, \tag{4}$$

where

$$A(\underline{x}, t) = \begin{pmatrix} t & \underline{f}^T \\ \underline{f} & 0 \end{pmatrix} + \sum_{i=1}^{37} x_i \begin{pmatrix} 0 & 0 \\ 0 & K_i \end{pmatrix}. \tag{5}$$

Having this semidefinite constraint the problem is formulated so that SeDuMi and SDPT-3 can solve it. Note that first, we are ignoring the constraints on nodal coordinates Y_j. The other constraint on cross-sectional areas is linear, so there will be no problem solving with SeDuMi and SDPT-3.

To optimise the nodal coordinates MATLAB built-in routine `fminsearch` is used. This time optimisation variables were the nodal coordinates and objective function the same as above. Note that $\underline{y} \in \Re^5$, $\underline{y} = \left[Y_3\ Y_5\ Y_7\ Y_9\ Y_{11}\right]$ since only upper nodes can have different locations and the structure is symmetric. Different results were obtained for different initial solutions using SeDuMi but almost the same using SDPT-3. The results obtained using SeDuMi are shown in Table 1 and using SDPT-3 in Table 2. The average number of iterations using SeDuMi was 46 and using SDPT-3 – 27, so SDPT-3 completed the program quicker. It can be observed from Table 1 that the minimal

Table 1. Results obtained with SeDuMi package

Initial y_0, m	[0.9 0.9 0.9 0.9 0.9]	[1 1 1 1 1]	[0.9 1 1.1 1.2 1.3]
$\lvert t \rvert$, Nm	620	680	680
Time, s	771	1001	382
Evaluations	709	1006	432
Iterations	301	401	217
Y_3, mm	492.2	518.7	1290.4
Y_5, mm	858.3	912.6	2067.2
Y_7, mm	1102.8	11.36	2554.4
Y_9, mm	1244.2	1296.7	2797.2
Y_{11}, mm	1323.6	1367.4	2844.2
$\lvert u_{10y} \rvert$, mm	8.28	9.30	9.31
Mass, kg	83.74	73.23	42.79

Table 2. Results obtained with SDPT-3 package

Initial y_0, m	[0.9 0.9 0.9 0.9 0.9]	[1 1 1 1 1]	[0.9 1 1.1 1.2 1.3]
$\lvert t \rvert$, Nm	620	680	680
Time, s	459	475	420
Evaluations	374	366	354
Iterations	242	217	215
Y_3, mm	1309.3	1289.2	1289.2
Y_5, mm	2100.4	2065.5	2065.5
Y_7, mm	2597.0	2552.9	2553.0
Y_9, mm	2844.0	2796.2	2769.2
Y_{11}, mm	2891.2	2843.3	2843.3
$\lvert u_{10y} \rvert$, mm	8.43	9.29	9.29
Mass, kg	42.53	42.79	42.79

mass of 42.79 kg is achieved when $\underline{y}_0 = [0.9\,1\,1.1\,1.2\,1.3]$, using different initial points mass is greater than 70 kg. SDPT-3 gives approximately 43 kg mass in all three cases. Thus SDPT-3 is more efficient for solving this problem.

Depending on initial solution and optimisation package, the obtained minimal masses fall into two different categories. One is of the mass approximately 43 kg, the other greater than 30 kg. However, the structure of mass 43 kg obtained using SeDuMi differs from the one of SDPT-3. To visualise these results see Figure 3.

The results obtained using two versions of particle swarm optimisation algorithm: improved PSO (Yancang *et al.* 2013) and PSO (Tang *et al.* 2009), can be found in Table 3. Both of the masses are much greater than the ones obtained using SDPT-3. The improved algorithm uses 25 particles and maximum number of iterations 1000. Therefore in total there were 25,000 iterations. The number of iterations obtained using `fminsearch` is difficult to determine because each function evaluation requires the other minimisation (this time using SDPT-3 or SeDuMi) which involves more iterations. Considering the number of iterations obtained using SDPT-3, when approximately 365 functions were evaluated and each evaluation completed approximately in 27 iterations, in total 9855 iterations were done. Under similar considerations the use of SeDuMi yields in total 19,136 iterations. Therefore, both

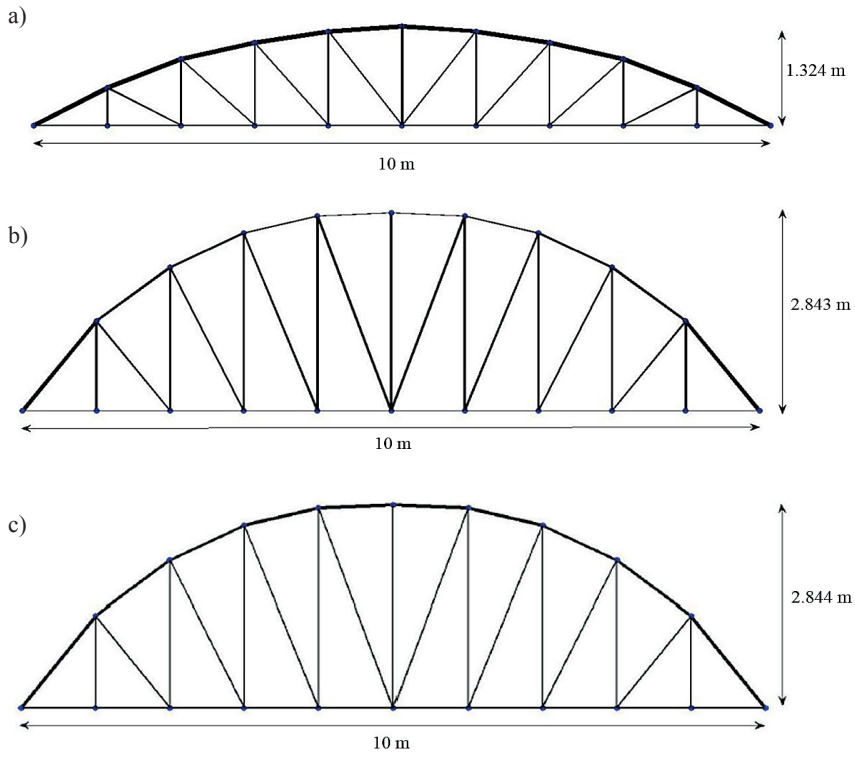

Fig. 3. Optimized structures: a – mass greater than 70 kg (SeDuMi), b – mass ~43 kg (SDPT-3), c – mass ~43 kg (SeDuMi)

of the methods use less iterations. However, the main disadvantage of solving such a problem using semidefinite programming comparing to PSO algorithm is that local minimisation is applied. Moreover, the initial point has to be chosen close to the optimal point, so that the constraints would be satisfied.

Table 3. Solutions using different algorithms

Variables	Algorithm	
	Improved PSO	PSO
Y_3, mm	496.5	508.2
Y_5, mm	897.6	904.4
Y_7, mm	1196.2	1178.1
Y_9, mm	1370.4	1346.1
Y_{11}, mm	1406.8	1363.4
$\lvert u_{10y} \rvert$, mm	9.93	8.07
Mass, kg	69.27	77.46

Conclusions

1. Semidefinite programming and packages SeDuMi and SDPT-3 suitable for using it were introduced in this paper.
2. The theory was applied to truss design optimisation problem.
3. The results using SDPT-3 were significantly better than the ones obtained using PSO algorithm. While the usage of SeDuMi was better depending on initial solution.

Nevertheless, the weakness of solving the problem in such a way is a highly restricted choice of initial solution and the fact that initial point had a huge impact on the final solution.

Acknowledgements

The work was supported by project "Promotion of Student Scientific Activities" (VP1-3.1-ŠMM-01-V-02-003) from the Research Council of Lithuania. This project is funded by the Republic of Lithuania and European Social Fund under the 2007–2013 Human Resources Development Operational Programme's priority 3.

References

Sturm, J. F. 1999. Using Sedumi 1.02, a Matlab toolbox for optimization over symmetric cones, *Optimization Methods and Software* 11(1–4): 625–653. http://dx.doi.org/10.1080/10556789908805766

Tang, H.; Li, F.; Wang, Y.; Xue, S.; Cheng, R. 2009. Particle swarm optimization algorithm for shape optimization of truss structures, *Journal of Harbin Institute of Technology* 41(12): 94–99.

Todd, M. J. 2001. Semidefinite optimization, *Acta Numerica* 10(5): 515–560. http://dx.doi.org/10.1017/S0962492901000071

Vandenberghe, L.; Boyd, S. 1996. Semidefinite programming, *SIAM Review* 38(1): 49–95. http://dx.doi.org/10.1137/1038003

Yancang, L.; Peng, Y.; Zhou, S. 2013. Improved PSO algorithm for shape and sizing optimization of truss structure, *Journal of Civil Engineering and Management* 19(4): 545–549. http://dx.doi.org/10.3846/13923730.2013.786754

SANTVAROS OPTIMIZAVIMO UŽDAVINIŲ SPRENDIMAS TAIKANT PUSIAU APIBRĖŽTĄ PROGRAMAVIMĄ

R. Giniūnaitė

Santrauka

Pusiau apibrėžtas programavimas yra iškiliojo optimizavimo posritis, kuriame tikslo funkcija tiesinė, o leistinoji sritis – pusiau teigiamai apibrėžtų matricų kūgio ir afininės erdvės sankirta. Tai gana naujas optimizavimo problemų sprendimo būdas, tačiau jau plačiai taikomas sprendžiant inžinerinius bei kombinatorinius optimizavimo uždavinius. Tokiems uždaviniams spręsti yra daug skirtingų paketų, taikančių įvairius algoritmus. Šiame darbe buvo naudojami *SeDuMi* ir *SDPT*-3 paketai, kuriuos, kaip ir daugumą kitų, galima parsisiųsti iš interneto. Tikslas buvo rasti minimalią santvaros masę atsižvelgiant į numatytus apribojimus. Naudojant *SDPT*-3 gauta optimali masė buvo vidutiniškai mažesnė nei naudojant *SeDuMi*. *SDPT*-3 veikė efektyviau ir pradinių sąlygų pasirinkimas neturėjo tokios didelės įtakos sprendiniui kaip naudojant *SeDuMi* paketą. Palyginus rezultatus su sprendiniais, gautais taikant dalelių spiečiaus optimizavimo algoritmą, nustatyta, kad tokio tipo uždaviniams pusiau apibrėžtas programavimas yra tinkamesnis.

Reikšminiai žodžiai: pusiau apibrėžtas programavimas, santvara, *SeDuMi*, *SDPT*-3.