CSE203B - Discussion Session

Po-Ya Hsu

02/05/21
Outline

• Convex Optimization
• CVX Basics
• Assignment Hints
Convex Optimization Problem in Standard Form

\[
\begin{align*}
\text{minimize} & \quad f_0(x) \\
\text{Subject to} & \quad f_i(x) \leq 0, i = 1, ..., m \\
& \quad a_i^T x = b_i, i = 1, ..., p
\end{align*}
\]

• The objective function \( f_0(x) \) must be convex
• The inequality constraints \( f_i(x) \) must be convex
• The equality functions \( h_i(x) = a_i^T x - b_i \) must be affine.
• The feasible set of a convex optimization problem is also convex.
Example: Max-Flow Problem

- Source: node 0
- Sink: node 5
- Objective: maximize the flow from the source to the sink
- Constraints: flow capacity, conservation of flow
Example: Max-Flow Problem

- Denote the graph as $G(V, E)$, flow as $f$, source as $s$, and sink as $t$
- Objective function: maximize $\sum_{v: <s, v> \in E} f(s, v)$
- Constraints:
  1. Capacity: $f(u, v) \leq c(u, v)$ & $f(u, v) \geq 0$ for all $v \in V - \{s, t\}$
  2. Conservation of flow
     $$\sum_{<u,v> \in E} f(u, v) = \sum_{<v,w> \in E} f(v, w), \text{ for all } v \in V - \{s, t\}$$
CVX Programming

• CVXPY: https://www.cvxpy.org/
CVX Basics

• To solve any convex optimization problem using CVX, your code should follow the structure shown below:

cvx_begin
    variable declaration
    objective function
    constraints
cvx_end
Revisit the Max-Flow Problem

• Find the solution using CVX
  1. Formulate the problem into a convex optimization problem (check slide 5 )
  2. Instantiate the problem data
  3. Construct the convex optimization problem following CVX structure
  4. Run the script
Revisit the Max-Flow Problem

• Find the solution using CVX
  1. Formulate the problem into a convex optimization problem (check slide 5)
  2. Instantiate the problem data
  3. Construct the convex optimization problem following CVX structure
  4. Run the script

```matlab
% capacity constraints
C = [16;13;10;4;12;9;14;7;20;4];

% conservation of flow
clf1 = [0,0,0,0,1,-1,0,1,-1,0];
clf2 = [0,0,0,0,0,-1,1,0,1];
clf3 = [0,1,1,-1,0,1,-1,0,0,0];
clf4 = [1,0,-1,1,-1,0,0,0,0,0];
```
Revisit the Max-Flow Problem

• Find the solution using CVX
  1. Formulate the problem into a convex optimization problem (check slide 5)
  2. Instantiate the problem data
  3. **Construct the convex optimization problem following CVX structure**
  4. Run the script

```matlab
% cvxopt
cvx_clear;
n = 10;
cvx_begin
  variable x(n)
  maximize( x(1) + x(2) )
  subject to
    -x <= 0;
    x <= C;
    cf1 * x == 0;
    cf2 * x == 0;
    cf3 * x == 0;
    cf4 * x == 0;

  cvx_end
```
Revisit the Max-Flow Problem

• Find the solution using CVX
  1. Formulate the problem into a convex optimization problem (check slide 5)
  2. Instantiate the problem data
  3. Construct the convex optimization problem following CVX structure
  4. Run the script

```matlab
>> run_max_flow
Calling SDPT3 4.0: 21 variables, 7 equality constraints
For improved efficiency, SDPT3 is solving the dual problem.

num. of constraints = 7
num. of linear var = 20
num. of free var = 1 *** convert able to blx
******************************************************************************
SDPT3: Infeasible path-following algorithm
******************************************************************************
version Predator gam eason scales_data
IT 1 0.000 0.0000
ct sdp em pinfeas dinfeas gap prnch-dk dual-dk opertime
0|0.0000|0.0001|4.7e-011|3.4e-000|7.1e03|0.000000e+03|1.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00
1|0.0000|0.0001|4.7e-011|3.4e-000|7.1e03|0.000000e+03|1.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00
2|0.0000|0.0001|4.7e-011|3.4e-000|7.1e03|0.000000e+03|1.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00
3|0.0000|0.0001|4.7e-011|3.4e-000|7.1e03|0.000000e+03|1.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00
4|0.0000|0.0001|4.7e-011|3.4e-000|7.1e03|0.000000e+03|1.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00
5|0.0000|0.0001|4.7e-011|3.4e-000|7.1e03|0.000000e+03|1.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00
6|0.0000|0.0001|4.7e-011|3.4e-000|7.1e03|0.000000e+03|1.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00
7|0.0000|0.0001|4.7e-011|3.4e-000|7.1e03|0.000000e+03|1.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00
8|0.0000|0.0001|4.7e-011|3.4e-000|7.1e03|0.000000e+03|1.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00
9|0.0000|0.0001|4.7e-011|3.4e-000|7.1e03|0.000000e+03|1.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00
10|0.0000|0.0001|4.7e-011|3.4e-000|7.1e03|0.000000e+03|1.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00
11|0.0000|0.0001|4.7e-011|3.4e-000|7.1e03|0.000000e+03|1.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00|0.000000e+00

stop: max(relative gap, infeasibilities) < 1.49e-08
------------------------------------------------------------------------
number of iterations = 11
primal objective value = -8.000000000e+00
dual objective value = -8.000000000e+00
gap := trace(XZ) = 1.99e-02
relative gap = 6.0e-10
actual relative gap = 1.92e-10
rel. primal infeas (scaled problem) = 4.00e-12
rel. dual = 0.00e+00
rel. primal infeas (unscaled problem) = 0.00e+00
rel. dual = 0.00e+00
norm(X), norm(y), norm(Z) = 2.2e+00, 1.2e+00, 3.6e+01
norm(A), norm(b), norm(C) = 7.5e+00, 3.0e+00, 4.9e+01
Total CPU time (secs) = 0.32
CPU time per iteration = 0.03
termination code = 0
SNDIT: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
------------------------------------------------------------------------
Status: Solved
Optimal value (cvx_optval) = -23
```
Assignment 1 - Hints

In a linear system,

\[ Y = \Phi X \]
\[ Y: d \times 1 \]
\[ \Phi: d \times S \]
\[ X: S \times 1 \]

When \( S>d \), the system is underdetermined. However, if the sparsity of the sources \( X \) is guaranteed, then compressed sensing can be applied to reconstruct the original sources.
Assignment 1 - Hints

In this assignment,

\[ Y = \Phi X + n \]

\( Y \): data
\( \Phi \): sinusoidal basis
\( X \): variable, sparsity guaranteed
\( n \): random noise

What you’ll have to do:
1) Formulate the convex optimization problem
2) Use CVX or CVXPY to solve \( X \)
3) Experiment with the weight in your objective function
Clarification

1) the basis functions $\Phi$: in general, when we consider a signal composed of $K$ sinusoids, we express as $y(t) = a_0 + \sum_{k=1}^{K} a_k e^{-i2\pi f_k t}$. However, in the assignment, the basis function is simplified to $\sum \sin(2\pi f_k t)$

2) the objective function: minimize the errors in the data fitting and enforce the sparsity, so it should look like

$$\minimize \alpha \| ? \|_1 + \| ? - ? \|_2^2$$

3) you’ll have the source as variables in your CVX programming

4) play with different weights $\alpha$
Assignments 2

• Clarification
  • The minimum volume ellipsoid problem is not an SDP, because the objective is not linear
  • However, the log determinant is a convex function. So it can still be solved using an SDP solver
Hints

• Decompose $R$ into $M^T M$ ($R = M^T M$)