Introduction to CVX

Olivier Fercoq and Rachael Tappenden

24 June 2014

4 - 5 pm, ICMS Newhaven Lecture Theatre
What is optimization?

Wikipedia: “an optimization problem consists of maximizing or minimizing a (real) function by systematically choosing input values from within an allowed set and computing the value of the function . . .

More generally, optimization includes finding ”best available” values of some objective function given a defined domain (or a set of constraints), including a variety of different types of objective functions and different types of domains.”
A bit of history

- Optimization has been going on for a long time. It dates back to at least Newton: (1669)
  - Newton’s method: root finding method $f(x) = 0$.
    $$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$
  - Above ‘iterative’ formulation is actually due to Simpson.
  - Simpson noted that the method can be used for solving optimization problems by setting the gradient to zero.
- Gaussian elimination – solving least squares problems
- Crowning achievement of calculus: differentiate a function, set the derivative to zero, giving the stationery points
More recent ‘history’

- In the last 20 years, progress in optimization has exploded.
- Computing power has advanced significantly.
- Lots of people think of OR as begin complicated, but it doesn’t have to be!
- Optimization is important in lots of modern day applications.
  - Signal processing
  - Image processing
  - Finance (Markowitz portfolio theory)
  - Energy (wind/wave power technology)
  - Digital economy (Google, database search engines)

Good news: optimization is absolutely everywhere
Bad news: can’t solve all optimization problems
More recent ‘history’

- In the last 20 years, progress in optimization has exploded.
- Computing power has advanced significantly
- Lots of people think of OR as begin complicated, but it doesn’t have to be!
- Optimization is important in lots of modern day applications.
  - Signal processing
  - Image processing
  - Finance (Markowitz portfolio theory)
  - Energy (wind/wave power technology)
  - Digital economy (Google, database search engines)

Good news: optimization is absolutely everywhere
Bad news: can’t solve all optimization problems (Exceptions?)
Convex optimization

Convex optimization is ‘nice’. (Any local solution to a convex optimization problem is also a **global** solution!)

Mathematical description:

\[
\min_x \quad f(x) \\
\text{subject to} \quad g(x) \leq 0 \\
\quad \quad h(x) = 0
\]

Where:

- The objective function \( f(x) \) is convex
- The inequality constraints are convex
- The equality constraints are affine
Convex vs nonconvex

\[ f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y) \]
What is cvx?

- CVX is a software package that runs in Matlab. It transforms Matlab into a modeling language for solving convex optimization problems.
- CVX is used to formulate and solve convex optimization problems.
- Type in a description of the problem (in Matlab) in a form that looks very similar to how one would write it mathematically on paper.
- CVX converts the problem description into an equivalent SDP and solves the problem (it calls a solver).
- Why is it so useful? We don’t need to worry about how CVX solves the problem!
What will I need to do?

- We need to ‘speak cvx’s language!’
  i.e., use the appropriate syntax so that cvx knows what we want it to do

- CVX will require you to follow some rules

- Learn a handful of rules to keep convexity

- Voluntarily accepting some restriction on the problems that can be solved, but then cvx will solve any problem you write down
Disciplined Convex Programming (DCP)

- Describe objective and constraints using expressions formed from
  - a set of basic atoms (convex, concave functions)
  - a restricted set of operations or rules (that preserve convexity)
- Modeling system keeps track of affine, convex, concave expressions
- Rules ensure that
  - expressions recognized as convex are convex
  - but, some convex expressions are not recognized as convex
- Problems described using DCP are convex by construction
CVX

- Uses DCP
- Download page and installation instructions: cvxr.com/cvx/download/
- Runs in Matlab, between cvx_begin and cvx_end
- Relies on SDPT3 or SeDuMi (LP/SOCP/SDP) solvers
- Refer to user guide, online help for more info: cvxr.com/cvx/
- The CVX example library has more than a hundred examples: cvxr.com/cvx/examples/
Example: Constrained norm minimization

\[
\min_x \|Ax - b\|_1 \\
\text{s.t} \quad -0.5 \leq x \leq 0.3
\]

A = randn(5, 3);
b = randn(5, 1);
cvx_begin
variable x(3);
minimize(norm(A*x - b, 1))
subject to
  -0.5 <= x;
x <= 0.3;
cvx_end

- Between `cvx_begin` and `cvx_end`, x is a CVX variable
- Statement `subject to` does nothing, but can be added for readability
- Inequalities are interpreted elementwise
What CVX does

After `cvx_end`, CVX

- Transforms problem into an LP
- Calls solver Sedumi
- Overwrites (object) `x` with (numeric) optimal value
- Assigns problem optimal value to `cvx_optval`
- Assigns problem status (which here is Solved) to `cvx_status`

(Had problem been infeasible, `cvx_status` would be Infeasible and `x` would be NaN)
Variables, objective and constraints

- Declare variables with
  variable name[(dims)] [attributes]
  - variables t x(8);
  - variable S(3,3) symmetric;
  - variable D(3,3) diagonal;

- Objective can be
  - minimize(convex expression)
  - maximize(concave expression)
  - omitted (feasibility problem)

- Constraints can be
  - convex expression <= concave expression
  - concave expression >= convex expression
  - affine expression == affine expression
  - omitted (unconstrained problem)
An example: portfolio optimization

- Modern portfolio theory (MPT): maximize portfolio expected return for a given amount of portfolio risk (or equivalent), by carefully selecting the proportions of various assets to invest in.

- Investing is a tradeoff between risk and expected return. In general, assets with higher expected return are riskier. For a given amount of risk, MPT describes how to select a portfolio with highest possible expected return. (Efficient frontier.)
The efficient frontier

The line represents The Efficient Frontier—the optimal combination of risk and return.

Each dot represents a portfolio. The dots that are closest to the Efficient Frontier line are the portfolios that are expected to show the best performance with the smallest risk.
Mathematical model

Several ways this problem can be modeled.

- Minimize the variance (risk), subject to a certain level of return:

  \[
  \min_x \ x^T \Sigma x
  \]
  subject to \( p^T x \geq p_{\text{min}} \).

- Maximize the profit, subject to a maximum level of risk:

  \[
  \max_x \ p^T x
  \]
  subject to \( x^T \Sigma x \leq r \).

Might have other constraints too; like no shorting, diversification, etc.
Example 1

Mathematical formulation:

Minimize risk, subject to achieving a certain profit,

\[
\begin{align*}
\min_{x} & \quad x^T \Sigma x \\
\text{s.t.} & \quad p^T x \geq p_{\text{min}} \\
& \quad x^T \mathbf{1} = 1 \\
& \quad x \geq 0
\end{align*}
\]

CVX formulation:

```matlab
load data
cvx_begin
variable x(n);
minimize (x'*Sigma*x);
subject to
    p'*x >= pmin;
    x'*ones(n,1) == 1;
    x >= 0;
cvx_end
cvx_optval
```
Mathematical formulation:

Maximize profit, subject to a fixed risk level,

\[
\max_x x^T p \\
\text{s.t} \quad x^T \Sigma x \leq r \\
x^T \mathbf{1} = 1 \\
x \geq 0
\]

CVX formulation:

```matlab
load data
cvx_begin
  variable x(n);
  maximize (x'*p);
  subject to
    x'*Sigma*x >= r;
    x'*ones(n,1) == 1;
    x >= 0;
  cvx_end
  cvx_optval
```
Composition rules

- $f$ convex and $a$ affine $\Rightarrow f(a)$ convex
  Example: $x \mapsto (b^T x - 2)^2$

- $f$ concave $\Rightarrow -f$ convex
  Example: $x \mapsto -\sqrt{x}$

- $h$ convex and nondecreasing and $g$ convex $\Rightarrow h(g)$ convex
  Example: $(x, y) \mapsto \exp(x^2 - y)$ is convex
  $(x, y) \mapsto (x^2 - y)^2$ is not convex
## Some functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Meaning</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>norm(x, p)</td>
<td>$|x|_p$</td>
<td>cvx</td>
</tr>
<tr>
<td>square(x)</td>
<td>$x^2$</td>
<td>cvx</td>
</tr>
<tr>
<td>square pos(x)</td>
<td>$(x_+)^2$</td>
<td>cvx, nondecr</td>
</tr>
<tr>
<td>pos(x)</td>
<td>$x_+$</td>
<td>cvx, nondecr</td>
</tr>
<tr>
<td>sum largest(x,k)</td>
<td>$x[1] + \ldots + x[k]$</td>
<td>cvx, nondecr</td>
</tr>
<tr>
<td>sqrt(x)</td>
<td>$\sqrt{x}$ (x $\geq$ 0)</td>
<td>ccv, nondecr</td>
</tr>
<tr>
<td>inv pos(x)</td>
<td>$1/x$ (x $&gt;$ 0)</td>
<td>cvx, nonincr</td>
</tr>
<tr>
<td>max(x)</td>
<td>$\max{x_1, \ldots, x_n}$</td>
<td>cvx, nondecr</td>
</tr>
<tr>
<td>quad over lin(x,y)</td>
<td>$x^2/y$ (y $&gt;$ 0)</td>
<td>cvx, nondecr in y</td>
</tr>
<tr>
<td>lambda max(X)</td>
<td>$\lambda_{\max}(X)(X = X^T)$</td>
<td>cvx</td>
</tr>
<tr>
<td>huber(x)</td>
<td>$\begin{cases} x^2, &amp;</td>
<td>x</td>
</tr>
</tbody>
</table>
Valid (recognized) examples

\[ u, v, x, y \text{ are scalar variables} \]
\[ X \text{ is a symmetric } 3 \times 3 \text{ variable} \]

- Convex:
  - \[ \|A\cdot x - y\| + 0.1\|x\|_1 \]
  - \[ \lambda_{\text{max}}(2\cdot X - 4\cdot \text{eye}(3)) \]
  - \[ \|2\cdot X - 3\|_{\text{fro}} \]

- Concave:
  - \[ \min(1 + 2\cdot u, 1 - \max(2, v)) \]
  - \[ \sqrt{v} - 4.55\cdot \text{inv\_pos}(u - v) \]
u, v, x, y are scalar variables

- Neither convex nor concave:
  - `square(x) - square(y)`
  - `norm(A*x - y) - 0.1*norm(x, 1)`

- Rejected due to limited DCP ruleset:
  - `sqrt(sum(square(x)))`
    (is convex; could use `norm(x)`)  
  - `square(1 + x^2)`
    (is convex; could use `square_pos(1 + x^2)` or `1 + 2*pow_pos(x, 2) + pow_pos(x, 4)`)

Rejected examples
Sets

- Some constraints are more naturally expressed with convex sets
- Sets in CVX work by creating unnamed variables constrained to the set
- Examples:
  - semidefinite(n)
  - nonnegative(n)
  - simplex(n)
  - lorentz(n)
- `semidefinite(n)`, say, returns an unnamed (symmetric matrix) variable that is constrained to be positive semidefinite
- `X == semidefinite(n)` means \( X \in S_n^+ \) (or \( X \succeq 0 \))
CVX hints/warnings

- Watch out for = (assignment) versus == (equality constraint)
- $X \geq 0$, with matrix $X$, is an elementwise inequality
- $X \geq \text{semidefinite}(n)$ means: $X$ is elementwise larger than some positive semidefinite matrix (which is likely not what you want)
- Writing subject to is unnecessary (but can look nicer)
- Use brackets around objective functions:
  use minimize $(c'\times x)$, not minimize $c'\times x$
Many problems stated using convex quadratic forms can be posed as norm problems (which can have better numerical properties):

\[ x'Px \leq 1 \] can be replaced with
\[ \text{norm}(\text{chol}(P)x) \leq 1 \]

- log, exp, entropy-type functions implemented using successive approximation method, which can be slow, unreliable
Useful references/resources

▶ http://cvxr.com/cvx/

Lab tomorrow from 11.30am - 12.30pm in Appleton tower. (Room M2a/M2b/M2c.)