# 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 $f(x) = 0$.

$$x_{k+1} = x_k - f(x_k)/f'(x_k)$$

    - 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:

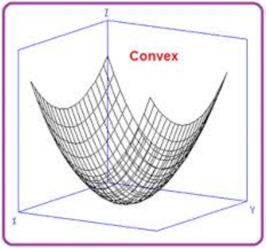
$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{subject to} \quad & g(x) \leq 0 \\ & h(x) = 0 \end{aligned}$$

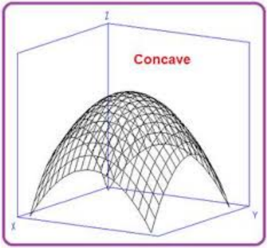
Where:

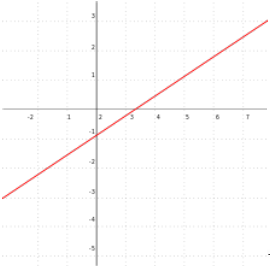
- The objective function $f(x)$ is convex
- The inequality constraints are convex
- The equality constraints are affine

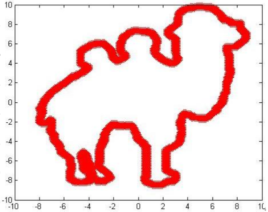
# Convex vs nonconvex

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y)$$





# 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

$$\begin{aligned} \min_x \quad & \|Ax - b\|_1 \\ \text{s.t} \quad & -0.5 \leq x \leq 0.3 \end{aligned}$$

```
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 intepreted 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`)

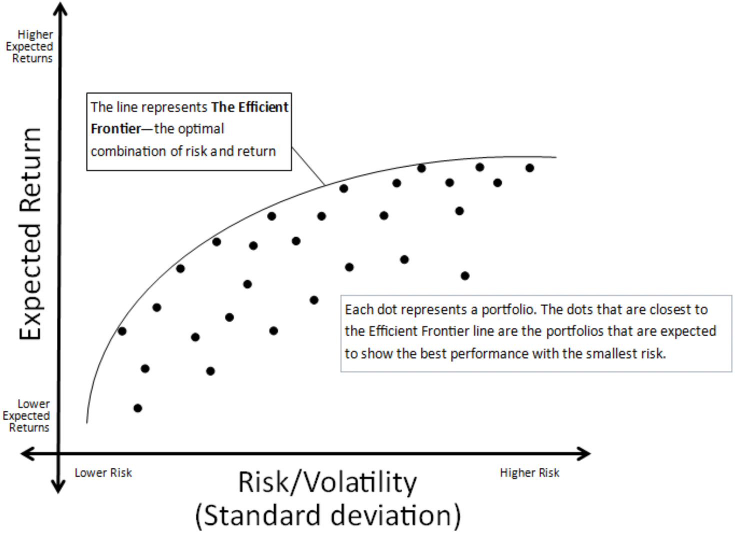
# Variables, objective and constraints

- 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



# Mathematical model

Several ways this problem can be modeled.

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

$$\begin{aligned} \min_x \quad & x^T \Sigma x \\ \text{subject to} \quad & p^T x \geq p_{\min}. \end{aligned}$$

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

$$\begin{aligned} \max_x \quad & p^T x \\ \text{subject to} \quad & x^T \Sigma x \leq r. \end{aligned}$$

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

# Example 1

Mathematical formulation:

Minimize risk, subject to achieving a certain profit,

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

CVX formulation:

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

# Example 2

Mathematical formulation:

Maximize profit, subject to a fixed risk level,

$$
\begin{aligned}
\max_{x} \quad & x^T p \\
\text{s.t} \quad & x^T \Sigma x \leq r \\
& x^T \mathbf{1} = 1 \\
& x \geq 0
\end{aligned}
$$

CVX formulation:

```
 load data
cvx_begin
  variable x(n);
  maximize (x'*p);
  subject to
    x'*Σ*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

| Function | Meaning | Attributes |
|---|---|---|
| `norm(x, p)` | $\|x\|_p$ | cvx |
| `square(x)` | $x^2$ | cvx |
| `square_pos(x)` | $(x_+)^2$ | cvx, nondecr |
| `pos(x)` | $x_+$ | cvx, nondecr |
| `sum_largest(x,k)` | $x_{[1]} + \ldots + x_{[k]}$ | cvx, nondecr |
| `sqrt(x)` | $\sqrt{x} \quad (x \geq 0)$ | ccv, nondecr |
| `inv_pos(x)` | $1/x \quad (x > 0)$ | cvx, nonincr |
| `max(x)` | $\max\{x_1, \ldots, x_n\}$ | cvx, nondecr |
| `quad_over_lin(x,y)` | $x^2/y \quad (y > 0)$ | cvx, nonincr in y |
| `lambda_max(X)` | $\lambda_{\max}(X)(X = X^T)$ | cvx |
| `huber(x)` | $\begin{cases} x^2, & \|x\| \leq 1 \\ 2\|x\| - 1, & \|x\| > 1 \end{cases}$ | cvx |

# Valid (recognized) examples

`u, v, x, y` are scalar variables

`X` is a symmetric 3 x 3 variable

- Convex:
  - `norm(A*x - y) + 0.1*norm(x, 1)`
  - `lambda_max(2*X - 4*eye(3))`
  - `norm(2*X - 3, fro)`
- Concave:
  - `min(1 + 2*u, 1 - max(2, v))`
  - `sqrt(v) - 4.55*inv_pos(u - v)`

# Rejected examples

`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)`)

# 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 $X \in \mathbf{S}^n_+$ (or $X \succeq 0$)

# CVX hints/warnings

- Watch out for = (assignment) versus == (equality constraint)
- `X >= 0`, with matrix `X`, is an elementwise inequality
- `X >= 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'*x)`, not `minimize c'*x`

# CVX hints/warnings (2)

- Many problems stated using convex quadratic forms can be posed as norm problems (which can have better numerical properties):
  `x'*P*x <= 1` can be replaced with
  `norm(chol(P)*x) <= 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.)