CME 252: Convex Sets, Functions, and Problems

AJ Friend
ICME, Stanford University

Set Notation

Outline

Set Notation

- $\mathbf{R}^n$: set of n-dimensional real vectors
- $x \in C$: the point x is an element of set C
- $C \subseteq \mathbf{R}^n$: C is a **subset** of $\mathbf{R}^n$, *i.e.*, elements of C are n-vectors
- can describe set elements explicitly: $1 \in \{3, \texttt{"cat"}, 1\}$
- **set builder notation**
$$C = \{x \mid P(x)\}$$
gives the points for which property $P(x)$ is true
- $\mathbf{R}^n_+ = \{x \mid x_i \geq 0 \text{ for all } i\}$: n-vectors with all nonnegative elements
- **set intersection**
$$C = \bigcap_{i=1}^{N} C_i$$
is the set of points which are simultaneously present in each C_i

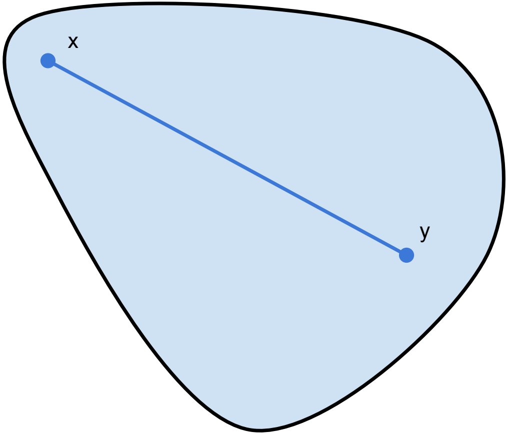
Convexity

Outline

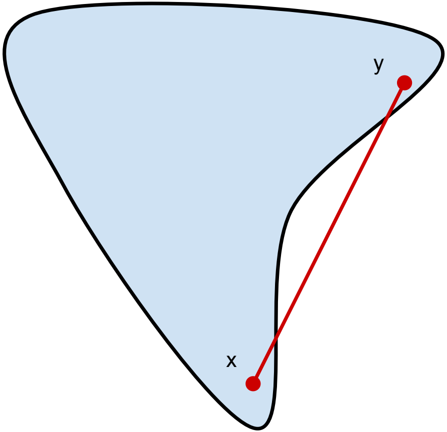
Convex Sets

- $C \subseteq \mathbf{R}^n$ is **convex** if
$$tx + (1-t)y \in C$$
for any $x, y \in C$ and $0 \leq t \leq 1$
- that is, a set is convex if the line connecting **any** two points in the set is entirely inside the set

Convex Set



Nonconvex Set



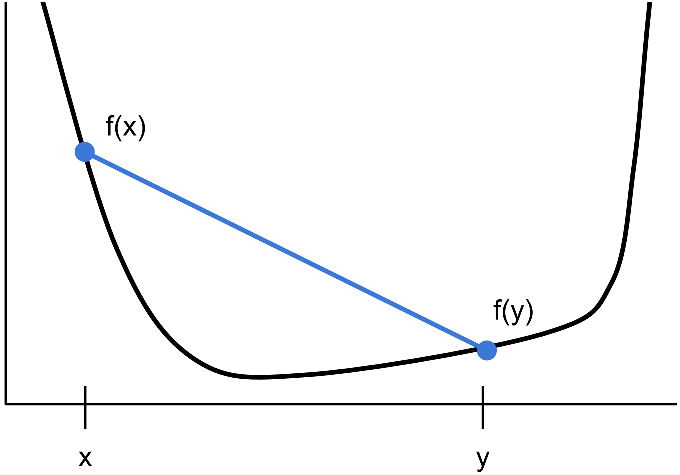
Convex Functions

- $f : \mathbf{R}^n \to \mathbf{R}$ is **convex** if $\mathbf{dom}(f)$ (the domain of f) is a convex set, and

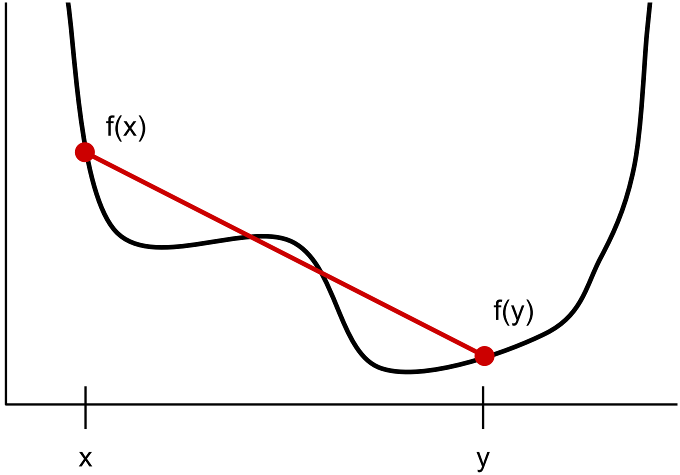
$$f\left(tx + (1-t)y\right) \leq tf(x) + (1-t)f(y)$$

  for any $x, y \in \mathbf{dom}(f)$ and $0 \leq t \leq 1$
- that is, convex functions are "bowl-shaped"; the line connecting any two points on the graph of the function stays above the graph
- f is **concave** if $-f$ is **convex**

Convex Function



Nonconvex Function



Convex Optimization Problem

- the optimization problem

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in C \end{array}$$

  is **convex** if $f : \mathbf{R}^n \to \mathbf{R}$ is convex and $C \subseteq \mathbf{R}^n$ is convex
- any **concave** optimization problem

$$\begin{array}{ll} \text{maximize} & g(x) \\ \text{subject to} & x \in C \end{array}$$

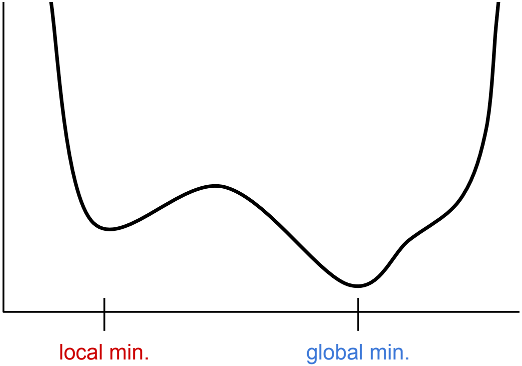
  for **concave** g and convex C can be rewritten as a **convex** problem by minimizing $-g$ instead

Why Convexity?

Outline

Minimizers

- no worries about local minimizers; local minimizers are global



Algorithms

- intuitive algorithms work: “just go down” leads you to the global minimum
- can’t get stuck close to local minimizers
- lots of good, existing software to solve convex optimization problems
- morally, writing down a convex optimization problem is as good as having the (computational) solution (for problems that aren’t too big!)

Expressiveness

- convexity is a modeling constraint; most problems are **not** convex
- however, convex optimization is **very** expressive, with many applications:
  - machine learning
  - engineering design
  - finance
  - signal processing
- convex modeling tools like CVX (MATLAB) or CVXPY (Python) make it easier to describe convex problems

Focus on Modeling

- modeling tools (CVX, CVXPY) and good solution algorithms let users (usually) focus on **what** their model should be instead of **how** to solve it
- coming up:
  - learn to manipulate **simple** convex sets and functions to **construct** more complicated convex models
  - theory of convex sets and functions
  - practical modeling with CVXPY

Nonconvex Extensions

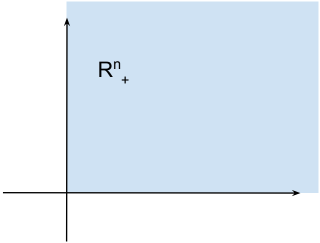
- even though most problems are not convex, convex optimization can still be useful
- approximate nonconvex problem with a convex model
- convex optimization can be used as a subroutine in a (heuristic) nonconvex solver:
    - locally approximate the problem as convex
    - solve local model
    - step to new point
    - re-approximate and repeat

Convex Sets

Outline

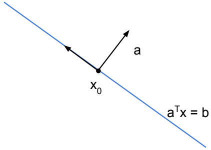
Examples

- empty set: $\emptyset$
- set containing a single point: $\{x_0\}$ for $x_0 \in \mathbf{R}^n$
- $\mathbf{R}^n$
- positive orthant: $\mathbf{R}^n_+ = \{x | x_i \geq 0, \ \forall i\}$

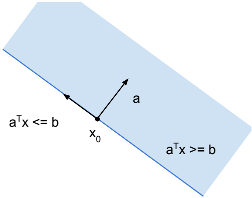


Hyperplanes and Halfspaces

- **hyperplane** $C = \{x | a^T x = b\}$



- **halfspace** $C = \{x | a^T x \geq b\}$



Norm Balls

- a norm $\|\cdot\| : \mathbf{R}^n \to \mathbf{R}$ is any function such that
  - $\|x\| \geq 0$, and $\|x\| = 0$ if and only if $x = 0$
  - $\|tx\| = |t|\|x\|$ for $t \in \mathbf{R}$
  - $\|x + y\| \leq \|x\| + \|y\|$
- $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$
- $\|x\|_1 = \sum_{i=1}^n |x_i|$
- $\|x\|_\infty = \max_i |x_i|$
- **unit norm ball**, $\{x | \|x\| \leq 1\}$, is **convex** for any norm

Norm Ball Proof

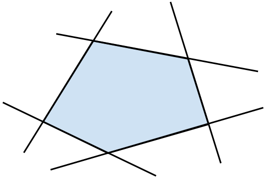
- let $C = \{x | \|x\| \leq 1\}$
- to check convexity, assume $x, y \in C$, and $0 \leq t \leq 1$
- then,

$$
\begin{aligned}
\|tx + (1-t)y\| &\leq \|tx\| + \|(1-t)y\| \\
&= t\|x\| + (1-t)\|y\| \\
&\leq t + (1-t) \\
&= 1
\end{aligned}
$$

- so $tx + (1-t)y \in C$, showing convexity
- this proof is typical for showing convexity

Intersection of Convex Sets

- the intersection of any number of convex sets is convex
- **example**: polyhedron is the intersection of halfspaces



- rewrite $\bigcap_{i=1}^{m}\{x|a_i^T x \leq b_i\}$ as $\{x|Ax \leq b\}$, where

$$A = \begin{bmatrix} a_1^T \\ \vdots \\ a_m^T \end{bmatrix}, \quad b = \begin{bmatrix} b_1^T \\ \vdots \\ b_m^T \end{bmatrix}$$

- $Ax \leq b$ is **componentwise** or **vector inequality**

More Examples

- solutions to a linear equation $Ax = b$ forms a convex set (intersection of hyperplanes)
- probability simplex, $C = \{x | x \geq 0, 1^T x = 1\}$ is convex (intersection of positive orthant and hyperplane)

CVXPY for Convex Intersection

- use CVXPY to solve the **convex set intersection problem**

$$\begin{array}{ll} \text{minimize} & 0 \\ \text{subject to} & x \in C_1 \cup \cdots \cup C_m \end{array}$$

- set intersection given by list of constraints
- **example**: find a point in the intersection of two lines

$$\begin{aligned} 2x + y &= 4 \\ -x + 5y &= 0 \end{aligned}$$

CVXPY code

```
from cvxpy import *

x = Variable()
y = Variable()

obj = Minimize(0)
constr = [2*x + y == 4,
          -x + 5*y == 0]

Problem(obj, constr).solve()

print x.value, y.value
```

- results in $x \approx 1.8$, $y \approx .36$

Diet Problem

- a classic problem in optimization is to meet the nutritional requirements of an army via various foods (with different nutritional benefits and prices) under cost constraints
- one soldier requires 1, 2.1, and 1.7 units of meat, vegetables, and grain, respectively, per day ($r = (1, 2.1, 1.7)$)
- one unit of hamburgers has nutritional value $h = (.8, .4, .5)$ and costs \$1
- one unit of cheerios has nutritional value $c = (0, .3, 2.0)$ and costs \$0.25
- prices $p = (1, 0.25)$
- you have a budget of \$130 to buy hamburgers and cheerios for one day
- can you meet the dietary needs of 50 soldiers?

Diet Problem

- write as optimization problem

$$
\begin{array}{ll}
\text{minimize} & 0 \\
\text{subject to} & p^T x \leq 130 \\
& x_1 h + x_2 c \geq 50r \\
& x \geq 0
\end{array}
$$

  with x giving units of hamburgers and cheerios
- or, with $A = [h, c]$,

$$
\begin{array}{ll}
\text{minimize} & 0 \\
\text{subject to} & p^T x \leq 130 \\
& Ax \geq 50r \\
& x \geq 0
\end{array}
$$

Diet Problem: CVXPY Code

```
x = Variable(2)
obj = Minimize(0)
constr = [x.T*p <= 130,
          h*x[0] + c*x[1] >= 50*r,
          x >= 0]

prob = Problem(obj, constr)
prob.solve(solver='SCS')
print x.value
```

- non-unique solution $x \approx (62.83, 266.57)$

Convex Functions

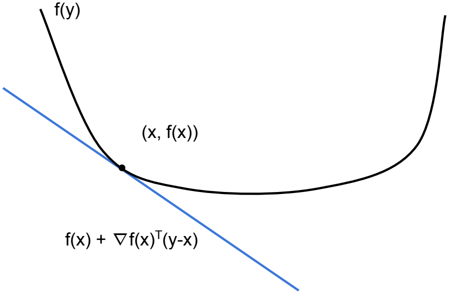
Outline

First-order condition

- for **differentiable** $f : \mathbf{R}^n \to \mathbf{R}$, the **gradient** ∇f exists at each point in $\mathbf{dom}(f)$
- f is convex if and only if $\mathbf{dom}(f)$ is convex and

$$f(y) \geq f(x) + \nabla f(x)^T (y - x)$$

  for all $x, y \in \mathbf{dom}(f)$
- that is, the first-order Taylor approximation is a **global underestimator** of f



Second-order condition

- for **twice differentiable** $f : \mathbf{R}^n \to \mathbf{R}$, the **Hessian** $\nabla^2 f$, or second derivative matrix, exists at each point in $\mathbf{dom}(f)$
- f is convex if and only if for all $x \in \mathbf{dom}(f)$,

$$\nabla^2 f(x) \succeq 0$$

- that is, the Hessian matrix must be **positive semidefinite**
- if $n = 1$, simplifies to $f^{''}(x) \geq 0$
- useful to determine convexity
- of course, there are many non-differentiable convex functions and the first- and second-order conditions generalize

Positive semidefinite matrices

- a matrix $A \in \mathbf{R}^{n \times n}$ is **positive semidefinite** $(A \succeq 0)$ if
  - A is **symmetric**: $A = A^T$
  - $x^T A x \geq 0$ for all $x \in \mathbf{R}^n$
- $A \succeq 0$ if and only if all **eigenvalues** of A are nonnegative
- intuition: graph of $f(x) = x^T A x$ looks like a bowl

Examples in **R**

$f(x)$	$f''(x)$
x	0
x^2	1
e^{ax}	$a^2 e^{ax}$
$1/x \ (x > 0)$	$2/x^3$
$-\log(x) \ (x > 0)$	$1/x^2$

Quadratic functions

- for $A \in R^{n \times n}$, $A \succeq 0$, $b \in \mathbf{R}^n$, $c \in \mathbf{R}$, the quadratic function

$$f(x) = x^T A x + b^T x + c$$

  is convex, since $\nabla^2 f(x) = A \succeq 0$
- in particular, the least squares objective

$$\|Ax - b\|_2^2 = x^T A^T A x - 2(Ab)^T x + b^T b$$

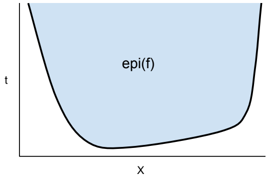
  is convex since $A^T A \succeq 0$

Epigraph

- the **epigraph** of a function is given by the set

$$\mathbf{epi}(f) = \{(x, t) \mid f(x) \leq t\}$$

- if f is convex, then $\mathbf{epi}(f)$ is convex



- the **sublevel sets** of a convex function

$$\{x \mid f(x) \leq c\}$$

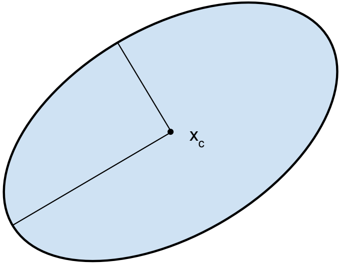
  are convex for any fixed $c \in \mathbf{R}$

Ellipsoid

- any **ellipsoid**

$$C = \{x \mid (x - x_c)^T P(x - x_c) \leq 1\}$$

with $P \succeq 0$ is convex because it is the sublevel set of a convex quadratic function



More convex and concave functions

- any norm is convex: $\|\cdot\|_1$, $\|\cdot\|_2$, $\|\cdot\|_\infty$
- $\max(x_1, \ldots, x_n)$ is convex
- $\min(x_1, \ldots, x_n)$ is concave
- absolute value $|x|$ is convex
- x^a is **convex** for $x > 0$ if $a \geq 1$ or $a \geq 0$
- x^a is **concave** for $x > 0$ if $0 \leq a \leq 1$
- **lots** more; for reference:
  - CVX Users' Guide, `http://web.cvxr.com/cvx/doc/funcref.html`
  - CVXPY Tutorial, `http://www.cvxpy.org/en/latest/tutorial/functions/index.html`
  - *Convex Optimization* by Boyd and Vandenberghe

Operations that preserve convexity

Positive weighted sums

- if $f_1, \ldots, f_n$ are convex and $w_1, \ldots, w_n$ are all positive (or nonnegative) real numbers, then
$$w_1 f_1(x) + \cdots + w_n f_n(x)$$
is also convex
- $7x + 2/x$ is convex
- $x^2 - \log(x)$ is convex
- $-e^{-x} + x^{0.3}$ is concave

Operations that preserve convexity

Composition with affine function

- if $f : \mathbf{R}^n \to \mathbf{R}$ is convex, $A \in \mathbf{R}^{n \times m}$, and $b \in \mathbf{R}^n$, then
$$g(x) = f(Ax + b)$$
is convex with $g : \mathbf{R}^m \to \mathbf{R}$
- mind the domain: $\mathbf{dom}(g) = \{x \mid Ax + b \in \mathbf{dom}(f)\}$

Operations that preserve convexity

Function composition

- let $f, g : \mathbf{R} \to \mathbf{R}$, and $h(x) = f(g(x))$
- if f is **increasing** (or nondecreasing) on its domain:
  - h is convex if f and g are convex
  - h is concave if f and g are concave
- if f is **decreasing** (or nonincreasing) on its domain:
  - h is convex if f is convex and g is concave
  - h is concave if f is concave and g is convex
- mnemonic:
  - "-" (decreasing) swaps "sign" (convex, concave)
  - "$+$" (increasing) keeps "sign" the same (convex, convex)

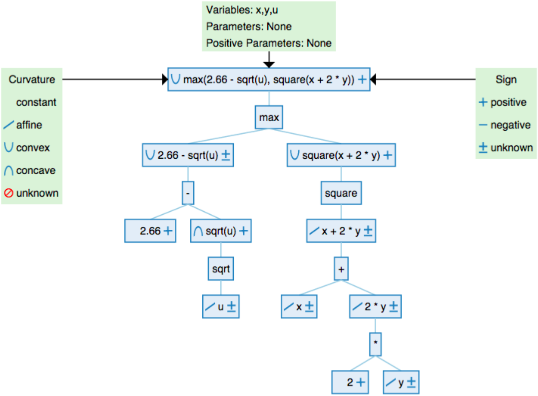
Operations that preserve convexity

Function composition examples

- mind the domain and range of the functions
- $\frac{1}{\log(x)}$ is convex (for $x > 1$)
  - $1/x$ is convex, decreasing (for $x > 0$)
  - $\log(x)$ is concave (for $x > 1$)
- $\sqrt{1 - x^2}$ is concave (for $|x| \leq 1$)
  - $\sqrt{x}$ is concave, increasing (for $x > 0$)
  - $1 - x^2$ is concave

Operations that preserve convexity

- dcp.stanford.edu website for constructing complex convex expressions to learn composition rules



CVXPY example

- recall that the **least squares** problem

$$\text{minimize} \quad \|Ax - b\|_2^2$$

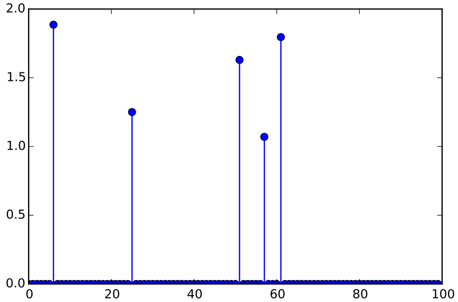
  is convex
- adding an $\|x\|_1$ term to the objective has an interesting effect: it "encourages" the solution x to be **sparse**
- the problem

$$\text{minimize} \quad \|Ax - b\|_2^2 + \rho\|x\|_1$$

  is called the LASSO and is central to the field of *compressed sensing*

CVXPY example

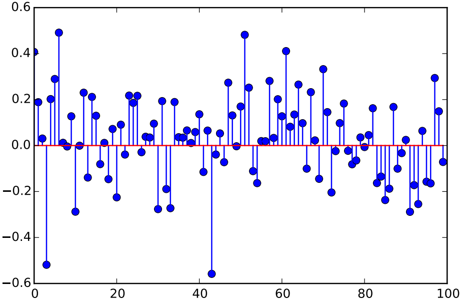
- $A \in \mathbf{R}^{30 \times 100}$, with $A_{ij} \sim \mathcal{N}(0, 1)$
- observe $b = Ax + \varepsilon$, where ε is noise
- more unknowns than observations!
- however, x is known to be sparse
- true x:



CVXPY example

least squares recovery given by

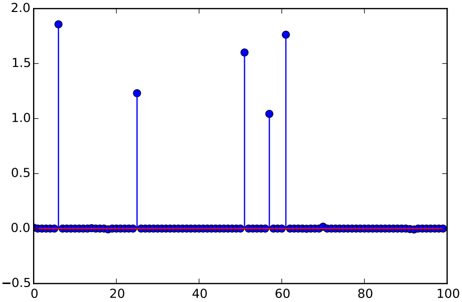
```
x = Variable(n)
obj = sum_squares(A*x - b)
Problem(Minimize(obj)).solve()
```



CVXPY example

LASSO recovery given by

```
x = Variable(n)
obj = sum_squares(A*x - b) + rho*norm(x,1)
Problem(Minimize(obj)).solve()
```



Convex Optimization Problems

Outline

Convex optimization problems

- combines convex objective functions with convex constraint sets
- constraints describe acceptable, or **feasible**, points
- objective gives desirability of feasible points

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in C_1 \\ & \vdots \\ & x \in C_n \end{array}$$

Constraints

- in CVXPY and other modeling languages, convex constraints are often given in epigraph or sublevel set form
  - $f(x) \leq t$ or $f(x) \leq 1$ for convex f
  - $f(x) \geq t$ for concave f

Equivalent problems

- loosely, we'll say that two optimization problems are **equivalent** if the solution from one is easily obtained from the solution to the other
- **epigraph** transformations:

$$\begin{array}{ll} \text{minimize} & f(x) + g(x) \end{array}$$

  equivalent to

$$\begin{array}{ll} \text{minimize} & t + g(x) \\ \text{subject to} & f(x) \le t \end{array}$$

Equivalent problems

- **slack variables**:

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax \le b \end{array}$$

  equivalent to

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax + t = b \\ & t \ge 0 \end{array}$$

Equivalent problems

- **dummy variables**:
$$\text{minimize} \quad f(Ax+b)$$
equivalent to
$$\begin{array}{ll} \text{minimize} & f(t) \\ \text{subject to} & Ax+b=t \end{array}$$

Equivalent problems

- **function transformations**:

$$\text{minimize} \quad \|Ax - b\|_2^2$$

equivalent to

$$\text{minimize} \quad \|Ax - b\|_2$$

since the square-root function is monotone

Examples

Outline

Diet problem

- wanted to know if we could feed an army of 50 with a budget of \$130:

$$
\begin{array}{ll}
\text{minimize} & 0 \\
\text{subject to} & p^T x \leq 130 \\
& x_1 h + x_2 c \geq 50 r \\
& x \geq 0
\end{array}
$$

  with x giving units of hamburgers and cheerios
- no objective; a set feasibility problem

Diet problem

- reformulate the problem to find the cheapest diet:

$$\begin{array}{ll} \text{minimize} & p^T x \\ \text{subject to} & x_1 h + x_2 c \geq 50r \\ & x \geq 0 \end{array}$$

- with CVXPY, we feed the troops for $129.17:

```
x = Variable(2)
obj = Minimize(x.T*p)
constr = [h*x[0] + c*x[1] >= 50*r,
          x >= 0]
Problem(obj, constr).solve()
```

Image in-painting

Original

Corrupted

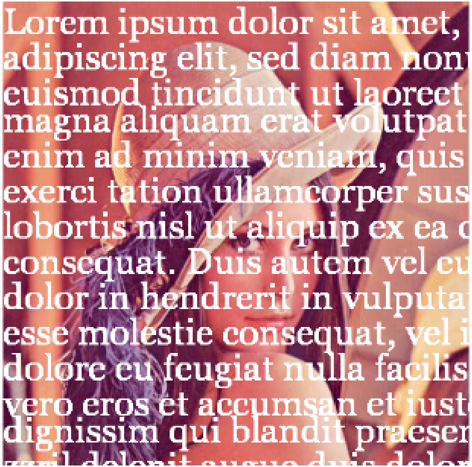


Image in-painting

guess pixel values in obscured/corrupted parts of image

- **decision variable** $x \in \mathbf{R}^{m \times n \times 3}$
- $x_{i,j} \in [0,1]^3$ gives RGB values of pixel (i,j)
- many pixels missing
- known pixel IDs given by set K, values given by **data** $y \in \mathbf{R}^{m \times n \times 3}$

total variation in-painting: choose pixel values $x_{i,j} \in \mathbf{R}^3$ to minimize

$$\mathsf{TV}(x) = \sum_{i,j} \left\| \begin{bmatrix} x_{i+1,j} - x_{i,j} \\ x_{i,j+1} - x_{i,j} \end{bmatrix} \right\|_2$$

that is, for each pixel, minimize distance to neighbors below and to the right, subject to known pixel values

In-painting: Convex model

$$
\begin{array}{ll}
\text{minimize} & \mathsf{TV}(x) \\
\text{subject to} & x_{i,j} = y_{i,j} \text{ if } (i,j) \in K
\end{array}
$$

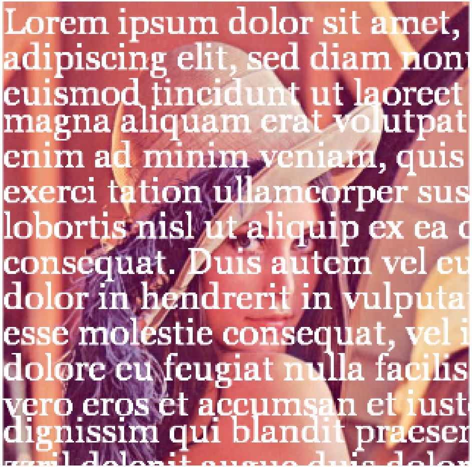
In-painting: Code example

```
# K[i, j] == 1 if pixel value known, 0 if unknown
from cvxpy import *
variables = []
constr = []
for i in range(3):
    x = Variable(rows, cols)
    variables += [x]
    constr += [mul_elemwise(K, x - y[:,:,i]) == 0]

prob = Problem(Minimize(tv(*variables)), constr)
prob.solve(solver=SCS)
```

In-painting: 512×512 color image; about 800k variables

Original

Corrupted



In-painting

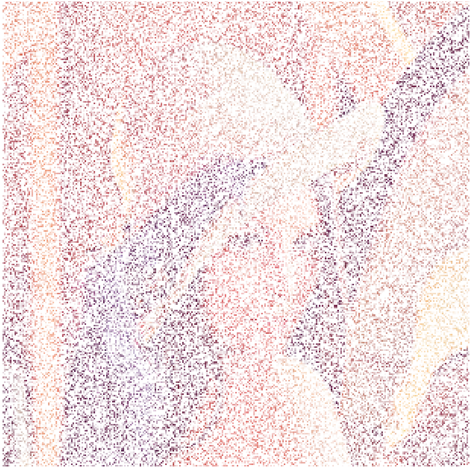
Original

Recovered

In-painting (80% of pixels removed)

Original

Corrupted

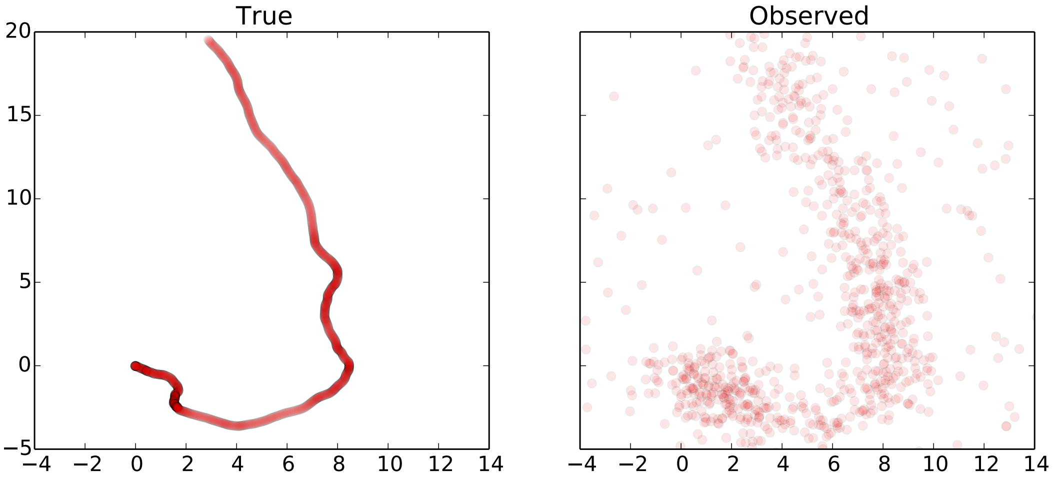


In-painting (80% of pixels removed)

Original

Recovered

Vehicle tracking



Kalman filtering

- estimate vehicle path from noisy position measurements (with outliers)
- dynamic model of vehicle state x_t:

$$x_{t+1} = Ax_t + Bw_t, \quad y_t = Cx_t + v_t$$

  - x_t is vehicle state (position, velocity)
  - w_t is unknown drive force on vehicle
  - y_t is position measurement; v_t is noise
- Kalman filter: estimate x_t by minimizing $\sum_t \left(\|w_t\|_2^2 + \gamma\|v_t\|_2^2\right)$
- a least-squares problem; assumes w_t, v_t Gaussian

Robust Kalman filter

- to handle outliers in v_t, replace square cost with Huber cost
- **robust** Kalman filter:

$$\begin{array}{ll} \text{minimize} & \sum_t \left(\|w_t\|_2^2 + \gamma\phi(v_t)\right) \\ \text{subject to} & x_{t+1} = Ax_t + Bw_t, \quad y_t = Cx_t + v_t \end{array}$$

where ϕ is Huber function

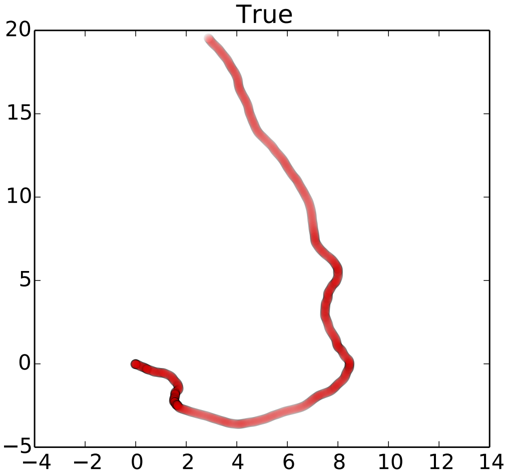
Robust KF CVXPY code

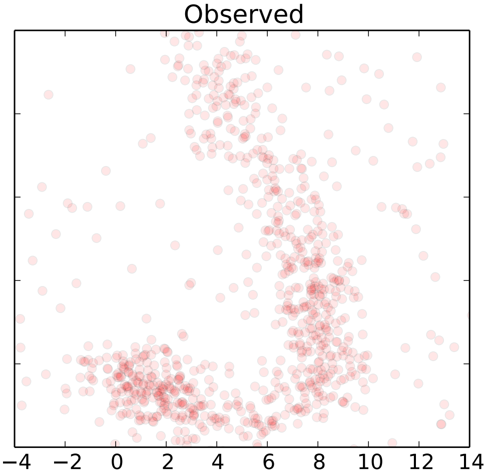
```
from cvxpy import *
x = Variable(4,n+1)
w = Variable(2,n)
v = Variable(2,n)

obj = sum_squares(w)
obj += sum(huber(norm(v[:,t])) for t in range(n))
obj = Minimize(obj)
constr = []
for t in range(n):
    constr += [ x[:,t+1] == A*x[:,t] + B*w[:,t] ,
                y[:,t]   == C*x[:,t] + v[:,t]   ]

Problem(obj, constr).solve()
```

Example

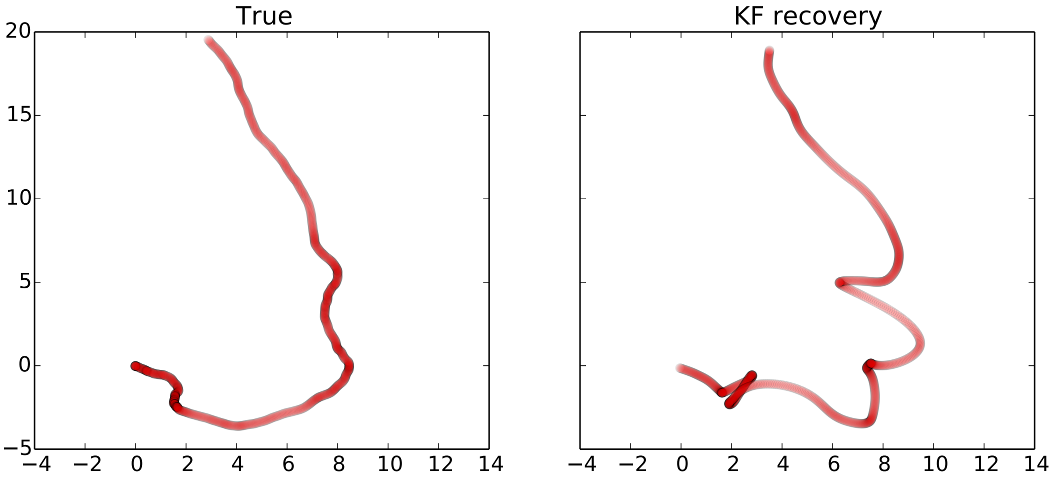
- 1000 time steps
- w_t standard Gaussian
- v_t standard Gaussian, except 30% are outliers with $\sigma = 10$

Example





Example



Example