

Convex-Concave Procedure

Stephen Boyd Maximilian Schaller Daniel Cederberg

Stanford University

Outline

Convex-concave procedure

Disciplined convex-concave programming

DCCP examples

Conclusions

Convex-concave procedure

the **convex-concave procedure** (CCP)

- ▶ is a heuristic method for solving a specific type of nonconvex problem
- ▶ involves solving a (typically small) sequence of convex problems
- ▶ leverages expressiveness and reliability of convex optimization
- ▶ is a good street-fighting trick to know about

Difference of convex functions

- ▶ a **difference of convex** (DC) function $h : \mathbf{R}^n \rightarrow \mathbf{R}$ has the form

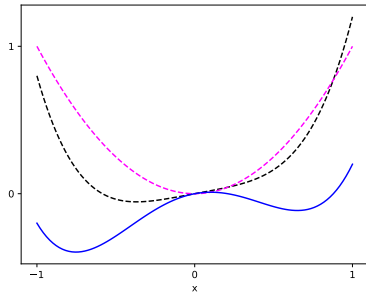
$$h(x) = f(x) - g(x)$$

with f and g convex

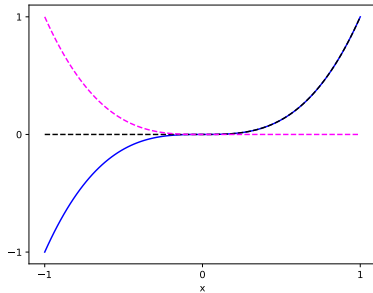
- ▶ any function with continuous second derivative has this form, but most useful when we have explicit expressions for f and g

Examples

$$f(x) = x^4 + 0.2x, \quad g(x) = x^2$$



$$f(x) = (x^3)_+, \quad g(x) = (x^3)_-$$



$f(x)$ in black, $g(x)$ in magenta, $h(x) = f(x) - g(x)$ in blue

Quadratic function as DC

- ▶ (nonconvex) quadratic function $h(x) = (1/2)x^T Px + q^T x + r$, $P \in \mathbf{S}^n$
- ▶ decompose P into its PSD and NSD parts $P = P_{\text{psd}} - P_{\text{nsd}}$, $P_{\text{psd}}, P_{\text{nsd}} \succeq 0$
 - $P = Q\Lambda Q^T$ is eigenvalue decomposition
 - $P_{\text{psd}} = Q\Lambda_+ Q^T$, with $\Lambda_+ = \max\{0, \Lambda\}$ (elementwise)
 - $P_{\text{nsd}} = Q\Lambda_- Q^T$, with $\Lambda_- = \max\{0, -\Lambda\}$ (elementwise)
- ▶ express h in DC form as $h = f - g$ with

$$f(x) = (1/2)x^T P_{\text{psd}}x + q^T x + r, \quad g(x) = (1/2)x^T P_{\text{nsd}}x$$

A simple majorizer for a DC function

- ▶ if (convex) g is differentiable, then for all x

$$\hat{g}(x; z) = g(z) + \nabla g(z)^T (x - z) \leq g(x)$$

(for nondifferentiable g , replace $\nabla g(x)$ with any subgradient)

- ▶ for DC function $h = f - g$, define

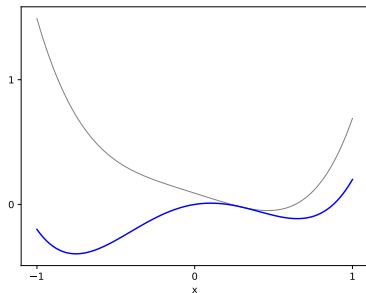
$$\hat{h}(x; z) = f(x) - \hat{g}(x; z)$$

- ▶ $\hat{h}$ is convex (in x) and satisfies $\hat{h}(x; z) \geq h(x)$ for all x , $\hat{h}(z; z) = h(z)$
- ▶ i.e., $\hat{h}(x; z)$ is a **majorizer** of h , tight at z

Examples

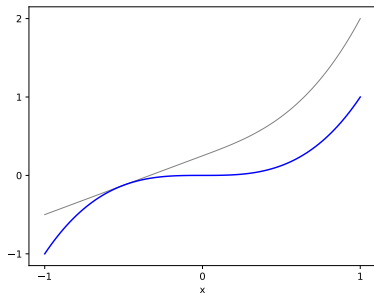
$$h(x) = x^4 + 0.2x - x^2,$$

majorized at $z = 0.3$



$$h(x) = (x^3)_+ - (x^3)_- = x^3,$$

majorized at $z = -0.5$



$h(x)$ in blue, $\hat{h}(x; z)$ in gray

Minimizing a DC function

- ▶ (unconstrained) **DC problem**: minimize DC function $h(x) = f(x) - g(x)$
- ▶ convex constraints can be handled as indicator functions added to f
- ▶ **convex-concave procedure** (CCP): iterate

$$x^{k+1} = \operatorname{argmin}_x \hat{h}(x; x^k) = \operatorname{argmin}_x \left(f(x) - \hat{g}(x; x^k) \right)$$

- ▶ x^{k+1} can be found via convex optimization
- ▶ CCP has no parameters, no line search / trust penalty, ...

Properties of CCP

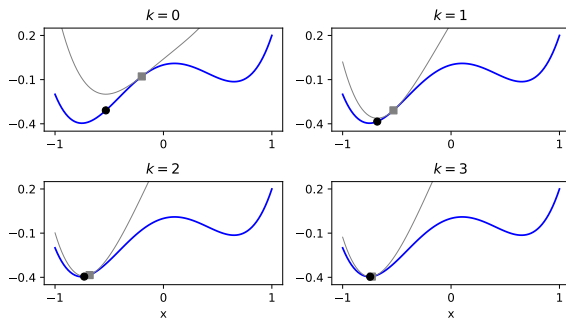
- ▶ CCP is a **descent method**:

$$h(x^{k+1}) \leq \hat{h}(x^{k+1}; x^k) \leq \hat{h}(x^k; x^k) = h(x^k)$$

- ▶ $h(x^k)$ converges, but not necessarily to $h^\star = \inf_x h(x)$
- ▶ ultimate value $\lim_{k \rightarrow \infty} h(x^k)$ depends on initial point x^0
- ▶ standard trick: run CCP for multiple initial points; take best ultimate point found

Example

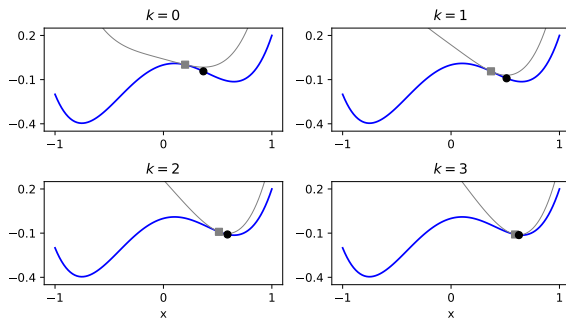
$f(x) = x^4 + 0.2x$, $g(x) = x^2$, initialized at $x^0 = -0.2$



$h(x)$ in blue, $\hat{h}(x; x^k)$ in gray,
 $(x^k, h(x^k))$ as squares, $(x^{k+1}, h(x^{k+1}))$ as circles

Example

$f(x) = x^4 + 0.2x$, $g(x) = x^2$, initialized at $x^0 = 0.2$



$h(x)$ in blue, $\hat{h}(x; x^k)$ in gray,
 $(x^k, h(x^k))$ as squares, $(x^{k+1}, h(x^{k+1}))$ as circles

SAT problem

- ▶ SAT problem: find $x_i \in \{0, 1\}$ with $Ax \preceq b$
- ▶ includes 3-SAT; NP-hard
- ▶ encode Boolean constraint using $x_i \in \{0, 1\} \iff x_i^2 - x_i = 0$
- ▶ SAT problem equivalent to minimizing DC function $h = f - g$ with

$$f(x) = \begin{cases} 0 & Ax \preceq b, 0 \preceq x \preceq \mathbf{1} \\ \infty & \text{otherwise} \end{cases} \quad g(x) = \sum_{i=1}^n (x_i^2 - x_i)$$

since $h(x) \geq 0$ for all x and

$$h(x) = 0 \iff Ax \preceq b, x_i \in \{0, 1\}, i = 1, \dots, n$$

SAT problem via CCP

- ▶ CCP: x^{k+1} is a solution of LP

$$\begin{array}{ll}\text{minimize} & (\mathbf{1} - 2x^k)^T x \\ \text{subject to} & Ax \preceq b, \quad 0 \preceq x \preceq \mathbf{1}\end{array}$$

- ▶ **example:** find $x \in \{0, 1\}^4$ that satisfies predicate

$$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_4)$$

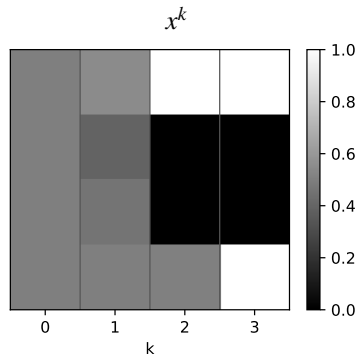
(a feasible 3-SAT instance with $n = 4$ variables and 3 clauses)

- ▶ can be represented as

$$A = \begin{bmatrix} -1 & -1 & 1 & 0 \\ 1 & 1 & 0 & -1 \\ 0 & 1 & -1 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

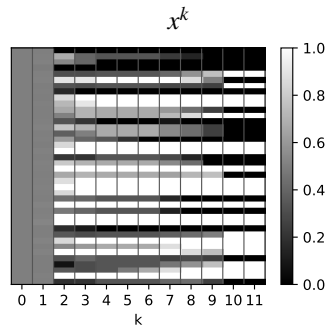
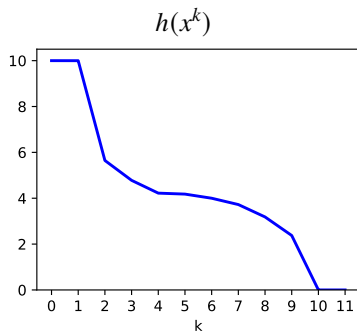
SAT problem via CCP

```
1 n = 4
2 x = Variable(n)
3 xk = Parameter(n)
4
5 obj = Minimize((ones(n) - 2 * xk) @ x)
6 constr = [A @ x <= b, 0 <= x, x <= 1]
7 prob = Problem(obj, constr)
8
9 xk.value = ones(n) / 2
10 for _ in range(3):
11     prob.solve()
12     xk.value = x.value
```



SAT problem via CCP

larger example, (feasible) 3-SAT instance with $n = 40$, 120 clauses



finds feasible point for 19% of random initializations

Constrained DC problem

- ▶ **DC problem** with DC inequality constraints has form

$$\begin{array}{ll}\text{minimize} & f_0(x) - g_0(x) \\ \text{subject to} & f_i(x) - g_i(x) \leq 0, \quad i = 1, \dots, m\end{array}$$

f_i and g_i are convex

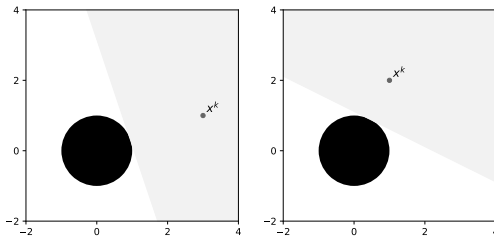
- ▶ CCP algorithm: x^{k+1} is solution of convex problem

$$\begin{array}{ll}\text{minimize} & f_0(x) - \hat{g}_0(x; x^k) \\ \text{subject to} & f_i(x) - \hat{g}_i(x; x^k) \leq 0, \quad i = 1, \dots, m\end{array}$$

- ▶ in words: linearize concave parts; solve; repeat
- ▶ if x^k is feasible, then so is x^{k+1} , and $h(x^{k+1}) \leq h(x^k)$
- ▶ CCP is a feasible descent method

Convex restrictions

- ▶ if $\hat{h}_i(x; x^k) = f_i(x) - \hat{g}_i(x; x^k) \leq 0$, then $h_i(x) \leq \hat{h}_i(x; x^k) \leq 0$
- ▶ so convexified constraint is a **convex restriction** of original constraint
- ▶ **example:** $f(x) = 0$, $g(x) = \|x\|_2 - 1$, $x^k = (3, 1)$ (left) and $x^k = (1, 2)$ (right)



$h(x) \leq 0$ in white, $\hat{h}(x; x^k) \leq 0$ in gray

Convex restrictions

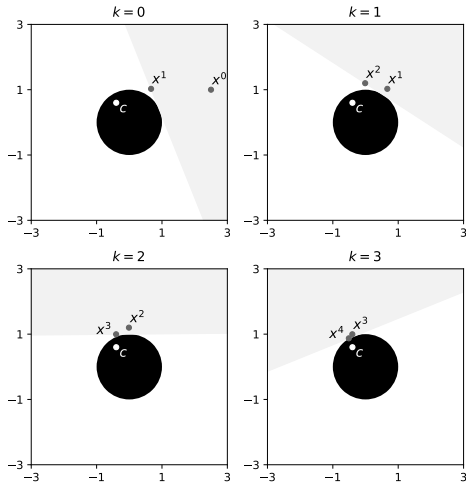
- ▶ **example:** CCP for simple problem

$$\begin{array}{ll}\text{minimize} & \|x - c\|_2 \\ \text{subject to} & \|x\|_2 \geq 1\end{array}$$

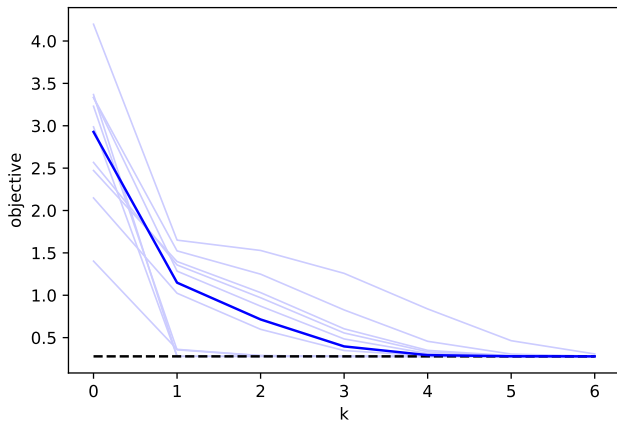
with variable $x \in \mathbf{R}^2$, data c with $\|c\|_2 < 1$, $c \neq 0$

- ▶ *i.e.*, find closest point to c that is **outside** unit ball
- ▶ solution is $x^\star = c/\|c\|_2$
- ▶ consider $c = (-0.4, 0.6)$ and $x^0 = (2.5, 1)$

CCP iterations



CCP iterations



$\|x^k - c\|_2$ in blue, $\|x^\star - c\|_2 = 1 - \|c\|_2$ as dashed black line

Outline

Convex-concave procedure

Disciplined convex-concave programming

DCCP examples

Conclusions

Disciplined convex-concave programming (DCCP)

- ▶ recall that a **DCP** problem is convex-by-construction, built from atoms combined by the DCP composition rules
- ▶ **disciplined convex-concave program** (DCCP) has form

$$\begin{array}{ll}\text{minimize/maximize} & o(x) \\ \text{subject to} & l_i(x) \sim r_i(x), \quad i = 1, \dots, m,\end{array}$$

- ▶ o, l_i, r_i are DCP convex or concave expressions
- ▶ $\sim$ can be $\leq$, $\geq$, or $=$
- ▶ to minimize DC objective $f_0 - g_0$
 - introduce epigraph variable t and DCCP constraint $f_0(x) - g_0(x) \leq t$
 - minimize t
- ▶ implemented in DCCP package

Disciplined convex-concave programming (DCCP)

► **example (revisited):**

$$\begin{array}{ll}\text{minimize} & \|x - c\|_2 \\ \text{subject to} & \|x\|_2 \geq 1\end{array}$$

with variable $x \in \mathbf{R}^2$, data c

```
1 x = Variable(2)
2 obj = Minimize(norm(x - c))
3 constr = [norm(x) >= 1]
4 prob = Problem(obj, constr)
5
6 x.value = [2.5, 1]
7 prob.solve(method='dccp')
8 print(x.value)
```

Outline

Convex-concave procedure

Disciplined convex-concave programming

DCCP examples

Conclusions

Path planning with obstacles

- ▶ find shortest path connecting points a and b in $\mathbf{R}^d$, avoiding m disks
- ▶ disk j has center c_j , radius r_j
- ▶ discretize path as points $x_0, \dots, x_n$ and solve

$$\begin{array}{ll}\text{minimize} & L \\ \text{subject to} & x_0 = a, \quad x_n = b \\ & \|x_i - x_{i-1}\|_2 \leq L/n, \quad i = 1, \dots, n \\ & \|x_i - c_j\|_2 \geq r_j, \quad i = 1, \dots, n, \quad j = 1, \dots, m\end{array}$$

with variables L and x

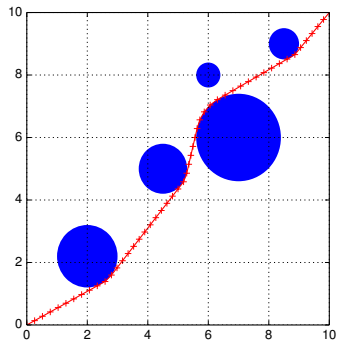
- ▶ useful initialization: straight line connecting a and b

Path planning with obstacles

```
1 L = Variable()
2 x = Variable((n+1, d))
3 L.value, x.value = ... # initialize to straight line
4
5 constr = [x[0] == a, x[n] == b]
6 for i in range(1, n+1):
7     constr += [norm(x[i] - x[i-1]) <= L/n]
8     for j in range(m):
9         constr += [norm(x[i] - c[j]) >= r[j]]
10
11 prob = Problem(Minimize(L), constr)
12 prob.solve(method='dccp')
```

Path planning with obstacles

$$d = 2, n = 50, m = 5$$



Outline

Convex-concave procedure

Disciplined convex-concave programming

DCCP examples

Conclusions

Conclusions

- ▶ the convex-concave procedure is a heuristic for difference of convex problems
 - linearize concave parts; solve; repeat
 - solves a (typically small) sequence of convex problems
- ▶ disciplined convex-concave programming automates CCP, via the DCCP package
- ▶ **CCP lacks optimality guarantees and depends on the initialization**