A DC programming framework for portfolio selection by minimizing the transaction costs

Pham Viet Nga

Department of Mathematics, Hanoi University of Agriculture

Table of contents

Table of contents

Problem description

- n assets: the return on asset i is a_i (random variable)
  $a = (a_1, \ldots, a_n), \quad \mathbf{E}(a) = \bar{a}, \quad \mathbf{E}(a - \bar{a})(a - \bar{a})^T = \Sigma.$
- $w = (w_i, \ldots, w_n)^T$: current holdings
- x_i: amount transacted in asset i,
  $x_i > 0$ for buying, $x_i < 0$ for selling,
  $x = (x_1, \ldots, x_n)^T$: portfolio selection.
- Adjusted portfolio $w + x$
  $\Longrightarrow$ The wealth at the end of the period: $W = a^T(w + x)$,
  $\mathbf{E}W = \bar{a}^T(w + x), \quad \mathbf{E}(W - \mathbf{E}W)^2 = (w + x)^T\Sigma(w + x)$
- Shortselling constraints: $w_i + x_i \geq -s_i, \ \forall i$ where $s_i \geq 0$
  Constraint on Expected return: $\bar{a}(w + x) \geq r_{\min}$
  Constraint on Variance: $(w + x)^T\Sigma(w + x) \leq \sigma^2_{\max}$
  Diversification constraints: $w_i + x_i \leq \lambda_i \mathbf{1}^T(w + x), \ \forall i, \ \lambda_i \geq 0$

Mathematical formulation

Problem (P)

$$\begin{cases} \min \phi(x) = \displaystyle\sum_{i=1}^{n} \phi_i(x_i) \\ \text{s.t. } \bar{a}(w+x) \geq r_{\min} \\ \qquad w_i + x_i \geq -s_i, \forall i \\ \qquad w_i + x_i \leq \lambda_i \mathbf{1}^T (w+x), \forall i \\ \qquad (w+x)^T \Sigma (w+x) \leq \sigma^2_{\max} \end{cases}$$

or $\min \left\{ \phi(x) = \sum_{i=1}^{n} \phi_i(x_i) \mid x \in C \right\}$

ϕ_i is *the transaction cost function for asset i* given by

$$\phi_i(x_i) = \begin{cases} 0, & x_i = 0 \\ \beta_i - \alpha_i^1 x_i, & x_i < 0 \\ \beta_i + \alpha_i^2 x_i, & x_i > 0 \end{cases} \qquad (\beta_i, \alpha_i^1, \alpha_i^2 \geq 0, \forall i)$$

Mathematical formulation

Problem (P)

$$\begin{cases} \min \phi(x) = \sum_{i=1}^{n} \phi_i(x_i) \\ \text{s.t. } \bar{a}(w+x) \geq r_{\min} \\ \quad w_i + x_i \geq -s_i, \forall i \\ \quad w_i + x_i \leq \lambda_i \mathbf{1}^T(w+x), \forall i \\ \quad (w+x)^T \Sigma (w+x) \leq \sigma_{\max}^2 \end{cases}$$

or $\min \left\{ \phi(x) = \sum_{i=1}^{n} \phi_i(x_i) \mid x \in C \right\}$

ϕ_i is *the transaction cost function for asset i* given by

$$\phi_i(x_i) = \begin{cases} 0, & x_i = 0 \\ \beta_i - \alpha_i^1 x_i, & x_i < 0 \\ \beta_i + \alpha_i^2 x_i, & x_i > 0 \end{cases} \qquad (\beta_i, \alpha_i^1, \alpha_i^2 \geq 0, \forall i)$$

Table of contents

DC programming and DCA: DC program

DC = Difference of Convex functions

DC program

$$(P_{dc}) \qquad \inf\{f(x) = g(x) - h(x) \mid x \in \mathbb{R}^n\}$$

(g, h : lower semicontinuous proper convex functions on $\mathbb{R}^n$)
$g - h$: a *DC decomposition* of f.
g, h : *convex DC components* of f.
g^*, h^* : *conjugate functions* of g, h.

$$g^*(y) := \sup\{\langle x, y\rangle - g(x) \mid x \in \mathbb{R}^n\}, \quad y \in \mathbb{R}^n$$

Subdifferential of a convex function θ at $x_0 \in \mathrm{dom} f$

($\mathrm{dom} f := \{x \in \mathbb{R}^n : \theta(x) \leq +\infty\}$)

$$\partial\theta(x_0) := \{y \in \mathbb{R}^n : \theta(x) \geq \theta(x_0) + \langle x - x_0, y\rangle, \forall x \in \mathbb{R}^n\}$$

DC programming and DCA: DC program

DC = Difference of Convex functions

DC program

$$(P_{dc}) \qquad \inf\{f(x) = g(x) - h(x) \mid x \in \mathbb{R}^n\}$$

(g, h : lower semicontinuous proper convex functions on $\mathbb{R}^n$)

$g - h$: a *DC decomposition* of f.

g, h : *convex DC components* of f.

g^*, h^* : *conjugate functions* of g, h.

$$g^*(y) := \sup\{\langle x, y\rangle - g(x) \mid x \in \mathbb{R}^n\}, \quad y \in \mathbb{R}^n$$

Subdifferential of a convex function θ at $x_0 \in \mathrm{dom} f$

$(\mathrm{dom} f := \{x \in \mathbb{R}^n : \theta(x) \leq +\infty\})$

$$\partial\theta(x_0) := \{y \in \mathbb{R}^n : \theta(x) \geq \theta(x_0) + \langle x - x_0, y\rangle, \forall x \in \mathbb{R}^n\}$$

DC programming and DCA: DC program

DC = Difference of Convex functions

DC program

$$(P_{dc}) \qquad \inf\{f(x) = g(x) - h(x) \mid x \in \mathbb{R}^n\}$$

(g, h : lower semicontinuous proper convex functions on $\mathbb{R}^n$)
$g - h$: a *DC decomposition* of f.
g, h : *convex DC components* of f.
g^*, h^* : *conjugate functions* of g, h.

$$g^*(y) := \sup\{\langle x, y\rangle - g(x) \mid x \in \mathbb{R}^n\}, \quad y \in \mathbb{R}^n$$

Subdifferential of a convex function θ at $x_0 \in \text{dom}f$

$(\text{dom}f := \{x \in \mathbb{R}^n : \theta(x) \leq +\infty\})$

$$\partial\theta(x_0) := \{y \in \mathbb{R}^n : \theta(x) \geq \theta(x_0) + \langle x - x_0, y\rangle, \forall x \in \mathbb{R}^n\}$$

DC programming and DCA: DC Algorithm

DCA = DC algorithm

Constructing two sequences $\{x^k\}$ et $\{y^k\}$, candidates to be solutions of (P_{dc}) and its dual program respectively

Generic DCA scheme

Initial point $x^0 \in \mathrm{dom} g,\ k \longleftarrow 0$

Repeat:

$$\begin{array}{ccc} & x^k \longrightarrow y^k \in \partial h(x^k) \\ & \swarrow \\ x^{k+1} \in \partial g^*(y^k) & \longrightarrow y^{k+1} \in \partial h(x^{k+1}) \end{array}$$

Until: convergence of $\{x^k\}$.

$x^{k+1} \in \partial g^*(y^k) \Longleftrightarrow$

$$x^{k+1} \in \arg\min\{g(x) - \langle x, y^k\rangle \mid x \in \mathbb{R}^n\}$$

BB: methods for global optimization for nonconvex problems

- *Lower bounds:* Found from convex relaxation, duality, Lipschitz or other bounds,...
- *Upper bounds:* can be found by choosing any point in the region, or by a local optimization method.
- *Basic idea:*
  - partition feasible set into convex sets, and find lower/upper bounds for each
  - form global lower and upper bounds; quit if close enough
  - else, refine partition and repeat

Table of contents

Method 1: Solving (P) by DCA

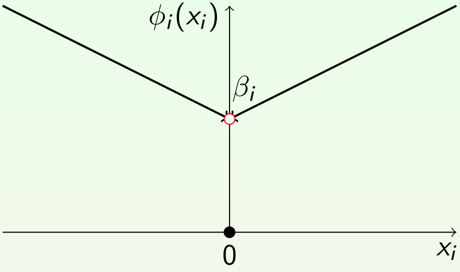
$$\phi_i(x_i) = \begin{cases} 0, & x_i = 0 \\ \beta_i - \alpha_i^1 x_i, & x_i < 0 \\ \beta_i + \alpha_i^2 x_i, & x_i > 0 \end{cases}$$

$$\phi(x) = \sum_{i=1}^{n} \phi_i(x_i)$$

$$f_i(x_i) = \begin{cases} \beta_i - \alpha_i^1 x_i, & x_i \le -\epsilon_i \\ -c_i^1 x_i, & -\epsilon_i \le x_i \le 0 \\ c_i^2 x_i, & 0 \le x_i \le \epsilon_i \\ \beta_i + \alpha_i^2 x_i, & x_i \ge \epsilon_i \end{cases}$$

$$(c_i^j = \frac{\beta_i}{\epsilon_i} + \alpha_i^j) \qquad f(x) = \sum_{i=1}^{n} f_i(x_i)$$

Method 1: Solving (P) by DCA



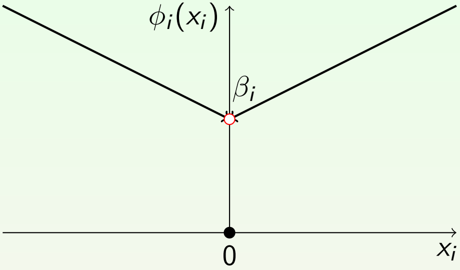
$$\phi_i(x_i) = \begin{cases} 0, & x_i = 0 \\ \beta_i - \alpha_i^1 x_i, & x_i < 0 \\ \beta_i + \alpha_i^2 x_i, & x_i > 0 \end{cases}$$

$$\phi(x) = \sum_{i=1}^{n} \phi_i(x_i)$$

$$f_i(x_i) = \begin{cases} \beta_i - \alpha_i^1 x_i, & x_i \le -\epsilon_i \\ -c_i^1 x_i, & -\epsilon_i \le x_i \le 0 \\ c_i^2 x_i, & 0 \le x_i \le \epsilon_i \\ \beta_i + \alpha_i^2 x_i, & x_i \ge \epsilon_i \end{cases}$$

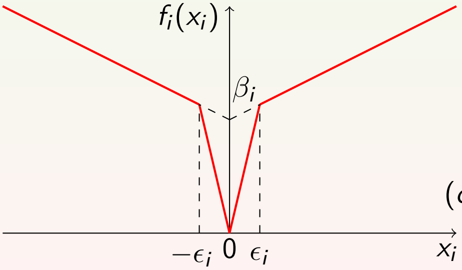
$$\left(c_i^j = \frac{\beta_i}{\epsilon_i} + \alpha_i^j\right) \qquad f(x) = \sum_{i=1}^{n} f_i(x_i)$$

Method 1: Solving (P) by DCA



$$\phi_i(x_i) = \begin{cases} 0, & x_i = 0 \\ \beta_i - \alpha_i^1 x_i, & x_i < 0 \\ \beta_i + \alpha_i^2 x_i, & x_i > 0 \end{cases}$$

$$\phi(x) = \sum_{i=1}^{n} \phi_i(x_i)$$



$$f_i(x_i) = \begin{cases} \beta_i - \alpha_i^1 x_i, & x_i \leq -\epsilon_i \\ -c_i^1 x_i, & -\epsilon_i \leq x_i \leq 0 \\ c_i^2 x_i, & 0 \leq x_i \leq \epsilon_i \\ \beta_i + \alpha_i^2 x_i, & x_i \geq \epsilon_i \end{cases}$$

$$(c_i^j = \frac{\beta_i}{\epsilon_i} + \alpha_i^j) \qquad f(x) = \sum_{i=1}^{n} f_i(x_i)$$

Method 1: Solving (P) by DCA

$$f(x) = \sum_{i=1}^{n} f_i(x_i)$$

- f is a polyhedral DC function
- $f(x) \le \phi(x), \forall x \in C$
- A DC approximation program of (P) is

$$(P_{dc}) \qquad \min\{f(x) = \sum_{i=1}^{n} f_i(x_i) = g(x) - h(x) \mid x \in C \cap R_0\}$$

- Solving (P_{dc}) by DCA to find a solution for (P)

(DCA has a finite convergence for polyhedral DC programs)

Method 2: Solving (P) by DCA-B&B

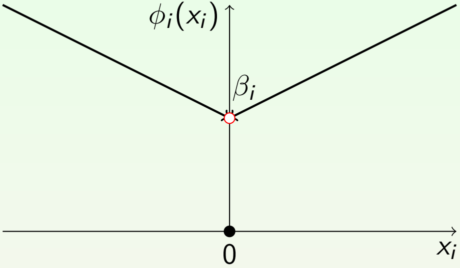
$$\phi_i(x_i) = \begin{cases} 0, & x_i = 0 \\ \beta_i - \alpha_i^1 x_i, & x_i < 0 \\ \beta_i + \alpha_i^2 x_i, & x_i > 0 \end{cases}$$

$$l_i^0 = \min\{x_i \mid x \in C\}$$

$$u_i^0 = \max\{x_i \mid x \in C\}$$

$$\widetilde{\phi}_i(x_i) = \begin{cases} \left(\frac{\beta_i}{l_i^0} - \alpha_i^1\right) x_i, & x_i \leq 0 \\ \left(\frac{\beta_i}{u_i^0} + \alpha_i^2\right) x_i, & x_i \geq 0 \end{cases}$$

Method 2: Solving (P) by DCA-B&B



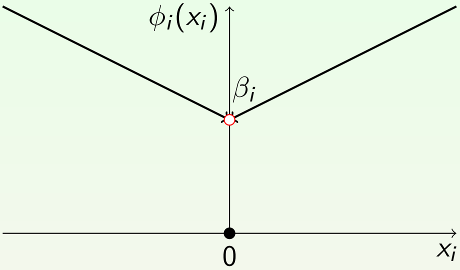
$$\phi_i(x_i) = \begin{cases} 0, & x_i = 0 \\ \beta_i - \alpha_i^1 x_i, & x_i < 0 \\ \beta_i + \alpha_i^2 x_i, & x_i > 0 \end{cases}$$

$$l_i^0 = \min\{x_i \mid x \in C\}$$

$$u_i^0 = \max\{x_i \mid x \in C\}$$

$$\widetilde{\phi}_i(x_i) = \begin{cases} \left(\frac{\beta_i}{l_i^0} - \alpha_i^1\right) x_i, & x_i \le 0 \\ \left(\frac{\beta_i}{u_i^0} + \alpha_i^2\right) x_i, & x_i \ge 0 \end{cases}$$

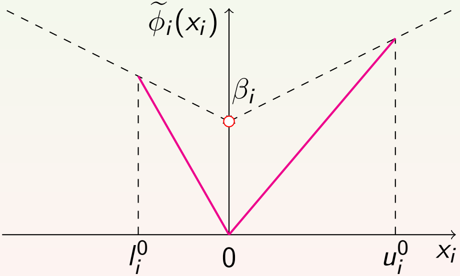
Method 2: Solving (P) by DCA-B&B



$$\phi_i(x_i) = \begin{cases} 0, & x_i = 0 \\ \beta_i - \alpha_i^1 x_i, & x_i < 0 \\ \beta_i + \alpha_i^2 x_i, & x_i > 0 \end{cases}$$

$$l_i^0 = \min\{x_i \mid x \in C\}$$

$$u_i^0 = \max\{x_i \mid x \in C\}$$



$$\widetilde{\phi}_i(x_i) = \begin{cases} \left(\frac{\beta_i}{l_i^0} - \alpha_i^1\right) x_i, & x_i \le 0 \\ \left(\frac{\beta_i}{u_i^0} + \alpha_i^2\right) x_i, & x_i \ge 0 \end{cases}$$

Method 2: Solving (P) by DCA-B&B

Iteration k:

$R_k = \{x \in R_0 \mid x_i = 0 \ \forall i \in I_k, \ x_j \neq 0 \ \forall j \in J_k\} \subset R_0 = [l^0, u^0]$

$I_k, J_k \subset \{1, 2, \ldots, n\}$ ($I_0 = \emptyset, J_0 = \emptyset$).

Subproblem of (P) at iteration k :

$$(P_k) \quad \omega_k = \min \left\{ \sum_{j \in J_k} \overline{\phi_j}(x_j) + \sum_{i \notin I_k \cup J_k} \phi_i(x_i) \mid x \in C \cap R_0, x_i = 0, \forall i \in I_k \right\}$$

Lower bound $\eta(R_k)$: relaxation problem $(R_k cp)$ of (P_k)

$$\eta(R_k) = \min \left\{ \sum_{j \in J_k} \overline{\phi_j}(x_j) + \sum_{i \notin I_k \cup J_k} \widetilde{\phi}_i(x_i) \mid x \in C \cap R_0, x_i = 0, \forall i \in I_k \right\}$$

Let x^{R_k} be a solution of $(R_k cp)$.

If $\phi(x^{R_k}) <$ the best upper bound $\Longrightarrow$ construct a DC approximation program of (P_k):

Method 2: Solving (P) by DCA-B&B

$$(R_k^{dc}) \qquad \min\left\{\left(\sum_{j\in J_k}\overline{\phi_j}(x_j)+\sum_{i\notin I_k\cup J_k}g_i(x_i)\right)-\left(\sum_{i\notin I_k\cup J_k}h_i(x_i)\right)\mid x\in C\cap R_0, x_i=0, \forall i\in I_k\right\}$$

Solving (R_k^{dc}) by DCA, let $x_{dc}^{R_k}$ be a solution.

Upper bound γ_k

$$\gamma_k=\min\{\phi(x^{R_k}),\phi(x_{dc}^{R_k})\}$$

Subdivision process

- *Choose an index i^* such that* $i^*\in\arg\max_i\{\phi_i(x_i^{R_k})-\widetilde{\phi}_i(x_i^{R_k})\}$
- $I_{k_1}:=I_k\cup\{i^*\}, J_{k_1}:=J_k$ et $I_{k_2}:=I_k, J_{k_2}:=J_k\cup\{i^*\}$
- $R_{k_1}=\{x\in R_k, x_{i^*}=0\}=\{x\in R_0\mid x_i=0\forall i\in I_{k_1}, x_j\neq 0\forall j\in J_{k_1}\}$
  $R_{k_2}=\{x\in R_k, x_{i^*}\neq 0\}=\{x\in R_0\mid x_i=0\forall i\in I_{k_2}, x_j\neq 0\forall j\in J_{k_2}\}$

Table of contents

Numerical results: Data

n assets: n^{th} is a riskless asset

The mean and covariance of $(n-1)$ risky assets were estimated from daily closing prices of $S\&P$ 500 stocks.

The mean of riskless asset is set to be 0.1.

$w_i = 1/n, \forall i = 1, \dots, n$

$\alpha_i^1 = \alpha_i^2 = \alpha_i = 0.01, \forall i = 1, \dots, n-1, \quad \alpha_n^1 = \alpha_n^2 = \alpha_n = 0$

$\beta_i = 0.1/(n-1),\ \forall i = 1, \dots, n-1, \quad \beta_n = 0$

$s_i = 5\beta_i,\ \forall i = 1, \dots, n-1, \quad s_n = 0.5$

$\lambda_i = 0.5,\ \forall i = 1, \dots, n.$

Tolerance to stop DCA: $\varepsilon = 10^{-8}$

The stopping criteria of BB algorithm (with DCA or without DCA) is either CPU time (in seconds) ≥ 1 hour or UB-LB $\leq 10^{-8}$

Numerical results

	BB			DCA (Algorithme 1)			DCA-BB (Algorithme 2)			
n	iter	Opt.val	CPU(s)	iter	valDCA	CPU(s)	iter	restart DCA	Opt.val	CPU(s)
101	4877	0.083683	1573.743	2	0.086818	0.374	4877	9	0.083683	1609.035
111	189	0.084263	51.995	2	0.084967	0.640	189	9	0.084263	56.690
121	316	0.084130	106.190	2	0.084734	0.765	316	8	0.084130	116.230
131	298	0.083862	101.924	2	0.084424	1.778	298	6	0.083862	113.374
141	495	0.083775	181.563	2	0.084254	0.936	495	3	0.083775	191.194
151	364	0.083678	139.508	2	0.084109	1.640	364	4	0.083678	142.824
161	612	0.083593	281.261	2	0.083982	1.912	612	4	0.083593	288.858
171	721	0.083516	519.326	2	0.083867	2.158	525	4	0.083516	428.666
181	986	0.083447	788.477	2	0.083765	2.586	563	3	0.083447	448.256
191	1135	0.083386	982.932	2	0.0836755	2.898	713	2	0.083386	759.186
201	1643	0.083329	1781.377	2	0.083592	2.839	751	2	0.083329	848.047

Table : Minimize the transaction costs

Numerical results

	BB			DCA (Algorithme 1)			DCA-BB (Algorithme 2)			
n	iter	Opt.val	CPU(s)	iter	valDCA	CPU(s)	iter	restart DCA	Opt.val	CPU(s)
101	4877	0.083683	1573.743	2	0.086818	0.374	4877	9	0.083683	1609.035
111	189	0.084263	51.995	2	0.084967	0.640	189	9	0.084263	56.690
121	316	0.084130	106.190	2	0.084734	0.765	316	8	0.084130	116.230
131	298	0.083862	101.924	2	0.084424	1.778	298	6	0.083862	113.374
141	495	0.083775	181.563	2	0.084254	0.936	495	3	0.083775	191.194
151	364	0.083678	139.508	2	0.084109	1.640	364	4	0.083678	142.824
161	612	0.083593	281.261	2	0.083982	1.912	612	4	0.083593	288.858
171	721	0.083516	519.326	2	0.083867	2.158	525	4	0.083516	428.666
181	986	0.083447	788.477	2	0.083765	2.586	563	3	0.083447	448.256
191	1135	0.083386	982.932	2	0.0836755	2.898	713	2	0.083386	759.186
201	1643	0.083329	1781.377	2	0.083592	2.839	751	2	0.083329	848.047

Table : Minimize the transaction costs

Numerical results

	BB			DCA (Algorithme 1)			DCA-BB (Algorithme 2)			
n	iter	Opt.val	CPU(s)	iter	valDCA	CPU(s)	iter	restart DCA	Opt.val	CPU(s)
101	4877	0.083683	1573.743	2	0.086818	0.374	4877	9	0.083683	1609.035
111	189	0.084263	51.995	2	0.084967	0.640	189	9	0.084263	56.690
121	316	0.084130	106.190	2	0.084734	0.765	316	8	0.084130	116.230
131	298	0.083862	101.924	2	0.084424	1.778	298	6	0.083862	113.374
141	495	0.083775	181.563	2	0.084254	0.936	495	3	0.083775	191.194
151	364	0.083678	139.508	2	0.084109	1.640	364	4	0.083678	142.824
161	612	0.083593	281.261	2	0.083982	1.912	612	4	0.083593	288.858
171	721	0.083516	519.326	2	0.083867	2.158	525	4	0.083516	428.666
181	986	0.083447	788.477	2	0.083765	2.586	563	3	0.083447	448.256
191	1135	0.083386	982.932	2	0.0836755	2.898	713	2	0.083386	759.186
201	1643	0.083329	1781.377	2	0.083592	2.839	751	2	0.083329	848.047

Table : Minimize the transaction costs

Thank you for attention!