مشتق های برداری

- $x=\begin{pmatrix}x_1\\x_2\\ \vdots\\x_n\end{pmatrix}$, $f(x): R^n \to R$, $\frac{\partial f}{\partial x}=\begin{pmatrix}\frac{\partial f}{\partial x_1}\\ \vdots\\ \frac{\partial f}{\partial x_n}\end{pmatrix}=f_x(x)=\underbrace{\nabla_x^T}_{\text{gradient}} f(x)$

- $\frac{\partial}{\partial x}(f_x(x))=f_{xx}(x)=\left[\frac{\partial^2 f}{\partial x_i x_j}\right]=[H_{ij}]=\begin{pmatrix}\frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_1}\\ \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n x_n}\\ \vdots & & \vdots\\ \frac{\partial^2 f}{\partial x_1 \partial x_n} & \cdots & \frac{\partial^2 f}{\partial^2 x_n}\end{pmatrix} \to$ Hessian Matrix

- $f\in C^2 \to$ ۲ بار مشتق پذیر پیوسته $\Rightarrow H=H^T$

$f\in C^1 \to$ ۱ « «

$f\in C \to$ تابع پیوسته

- $f(x,y)=y^TAx \to \frac{\partial f}{\partial x}=A^Ty$

اثبات:

$y^TAx=[y_1,\dots,y_n]\begin{bmatrix}a_{11} & \cdots & a_{1n}\\ \vdots & & \vdots\\ a_{n1} & \cdots & a_{nn}\end{bmatrix}\begin{bmatrix}x_1\\ \vdots\\ x_n\end{bmatrix}=\sum_{j=1}^{n}\sum_{i=1}^{n} y_i a_{ij} x_j=\sum_{j=1}^{n} y^T a_j x_j$

$\frac{\partial}{\partial x_k}(y^TAx)=\frac{\partial}{\partial x_k}\left(\sum_j y^T a_j x_j\right)=y^T a_k$

$\Rightarrow \frac{\partial}{\partial x}(y^TAx)=\begin{pmatrix}y^Ta_1\\ \vdots\\ y^Ta_n\end{pmatrix}=\begin{pmatrix}a_1^Ty\\ \vdots\\ a_n^Ty\end{pmatrix}=\begin{pmatrix}a_1^T\\ \vdots\\ a_n^T\end{pmatrix}y=A^Ty$

- $\frac{\partial(Ax)}{\partial x}=\frac{\partial}{\partial x}\begin{pmatrix}\sum_{i=1}^{n} a_{1i}x_i\\ \vdots\\ \sum_{i=1}^{n} a_{ni}x_i\end{pmatrix}=\begin{pmatrix}a_{11} & \cdots & a_{n1}\\ a_{12} & \cdots & a_{n2}\\ \vdots & & \vdots\\ a_{1n} & \cdots & a_{nn}\end{pmatrix}=A^T$

- $\frac{\partial y_{m\times1}(x)}{\partial x_{n\times1}}=\begin{pmatrix}\frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1}\\ \vdots & & \vdots\\ \frac{\partial y_1}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n}\end{pmatrix}_{n\times m}$

- $\frac{\partial}{\partial x}(x^TAx)=\underbrace{A^Tx}_{x^T \text{ ثابت فرض شده}}+\underbrace{Ax}_{Ax \text{ ثابت فرض شده}}$ $\qquad x^TAx=(x^TAx)^T=x^TA^Tx \xrightarrow[\text{در حالتی } x^T \text{ ثابت باشد}]{\text{مشتق نسبت به } x} = Ax$

باید یک بار Ax ثابت فرض شود و مشتق گرفته شود و دفعه دیگر
x^T ثابت فرض شود و مشتق گرفته شود.

بسط تیلور تابع اسکالر حول x^*:

$f(x)=f(x^*)+f_x^T(x^*)(\overbrace{x-x^*}^{\Delta x})+\frac{1}{2!}(\overbrace{x-x^*}^{\Delta x})^T H(x^*)(x-x^*)+HoT$

$f_x^T(x^*)=\left[\frac{\partial f}{\partial x_1} \cdots \frac{\partial f}{\partial x_n}\right]\Big|_{x=x^*}$

$=\frac{1}{2}\frac{\partial^2 f}{\partial x_1^2}\Delta x_1^2+\frac{1}{2}\frac{\partial^2 f}{\partial x_1 \partial x_2}\Delta x_1 \Delta x_2 + \cdots + \frac{1}{2}\frac{\partial^2 f}{\partial x_n^2}\Delta x_n^2$

Subject :
Year . Month . Date .

$$x - x^* = \Delta x \;,\quad x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} ,\; x^* = \begin{pmatrix} x_1^* \\ \vdots \\ x_n^* \end{pmatrix} \to \Delta x = \begin{pmatrix} x_1 - x_1^* \\ \vdots \\ x_n - x_n^* \end{pmatrix} = \begin{pmatrix} \Delta x_1 \\ \vdots \\ \Delta x_n \end{pmatrix}$$

تعیین علامت ماتریس:

ماتریس های معین (نیمه معین) مثبت و منفی: یک ماتریس را نمی‌توان تعیین علامت کرد چون ماتریس از درایه های مختلف تشکیل شده و هم مثبت و منفی هم برای اسکالر تعریف می‌شود؛ حال یک اسکالری که در مسائل بهینه سازی به آن برمی‌خوریم

x^TAx می‌باشد و در واقع می‌خواهیم این را تعیین علامت کنیم! $x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} , A = \begin{bmatrix} a_{11} & \dots & a_{n1} \\ \vdots & & \vdots \\ a_{m} & \dots & a_{nn} \end{bmatrix} = \sum a_{ij} x_i x_j$

تعریف: ماتریس حقیقی $A_{n\times n}$ معین مثبت (نیمه معین مثبت) است اگر:

$\forall x \in R^n , x \neq 0 : x^TAx \geq 0$

معین منفی (نیمه معین منفی) بودن ماتریس A معادل معین مثبت (نیمه معین مثبت) بودن ماتریس $-A$ است.

قضیه: شرط لازم کافی برای معین مثبت بودن (نیمه معین مثبت بودن) ماتریس $A_{n\times n}$ این است که قسمت متقارن ماتریس A معین مثبت (نیمه معین مثبت) باشد:

اثبات:

$$A = \underbrace{\frac{1}{2}(A+A^T)}_{\text{قسمت متقارن}} + \frac{1}{2}(A-A^T)$$

$$x^TAx = \frac{1}{2}x^T(A+A^T)x + \underbrace{\frac{1}{2}x^T(A-A^T)x}_{\text{اسکالر} = C} \qquad C = C^T = \frac{1}{2}x^T(A^T-A)x = -C$$

$$\Rightarrow C = -C \Rightarrow \boxed{C=0}$$

$$\Rightarrow x^TAx = x^T\left(\frac{A+A^T}{2}\right)x$$

قضیه: هر ماتریس متقارن قطری شدنی است. اثبات:

فرض کنیم ماتریس متقارن M، بردار ویژه تعمیم یافته از مرتبه ۲ را داشته باشد:

بردار ویژه تعمیم یافته از مرتبه ۲ متعلق به مقدار ویژه λ از ماتریس متقارن M است :

$$\begin{cases} (M-\lambda I)x \neq 0 \\ (M-\lambda I)^2 x = 0 \end{cases}$$

$$[(M-\lambda I)x]^T[(M-\lambda I)x] = y^T y \neq 0 \rightarrow$$ چون y خودش صفر نیست

$$= x^T(M-\lambda I)^T(M-\lambda I)x = x^T(M^T-\lambda I)(M-\lambda I)x = x^T(M-\lambda I)^2 x = 0 \quad \Big\} \Rightarrow 0 \neq 0 \Rightarrow \times$$

بنابراین ماتریس متقارن، بردار ویژه تعمیم یافته از مرتبه ۲ ندارد. بدین ترتیب ماتریس متقارن بردار ویژه تعمیم یافته از مراتب بالاتر نیز نخواهد داشت و در نتیجه همه بردار ویژه‌های ماتریس متقارن از مرتبه ۱ اند و بنابراین ماتریس متقارن قطری شدنی است.

قضیه: مقادیر ویژه ماتریس متقارن حقیقی هستند.

اثبات: فرض کنیم ماتریس متقارن M، مقدار ویژه‌ای به صورت $\lambda_i = \alpha_i + j\beta_i$ داشته باشد و v_i بردار ویژه متناظر با آن باشد.

$$M v_i = (\alpha_i + j\beta_i) v_i \xrightarrow{\times v_i^{*T}} v_i^{*T} M v_i = v_i^{*}(\alpha_i + j\beta_i) v_i$$

$$\big\downarrow {*}\text{ and } T$$

$$v_i^{*T} M^T = (\alpha_i - j\beta_i) v_i^{*} \xrightarrow{\times v_i} v_i^{*T} M^T v_i = (\alpha_i - j\beta_i) v_i^{*} v_i$$

$$\Rightarrow (\alpha_i - j\beta_i)\,\|v_i\|^2 = (\alpha_i + j\beta_i)\,\|v_i\|^2 \Rightarrow -\beta_i = \beta_i \Rightarrow \beta_i = 0$$

قضیه: بردارهای ویژه ماتریس متقارن برهم عمودند، Q: ماتریس بردارهای ویژه ; اثبات:

$$\Lambda = Q^{-1} M Q$$

$$M^T = M = Q\Lambda Q^{-1} = (Q^{-1})^T \Lambda Q^T \Rightarrow \Lambda = Q^{-1}(Q^{-1})^T \Lambda Q^T Q$$

$$\Lambda = \Lambda^T = Q^T Q \Lambda (Q^{-1})(Q^{-1})^T = Q^T Q \Lambda (Q^T Q)^{-1} = P \Lambda P^{-1}$$

P باید قطری باشد:

$$P = \begin{bmatrix} \alpha_1 & \cdots & 0 \\ \vdots & \alpha_2 & \vdots \\ 0 & \cdots & \alpha_n \end{bmatrix} = Q^T Q, \quad \begin{bmatrix} q_1^T \\ \vdots \\ q_n^T \end{bmatrix} [q_1 \cdots q_n] \rightarrow \begin{cases} i \neq j & q_i^T q_j = 0 \\ i = j & q_i^T q_j = \alpha_i > 0 \end{cases} \quad (*)$$

Subject :
Year . Month . Date .

بردارهای ویژه ماتریس متقارن دو به دو بر هم عمودند. اگر بردارهای ویژه را نرمال کنیم بطوری که اندازه آنها برابر ۱ باشد خواهیم داشت: $\|q_i\|^2 = q_i^T q_i = 1 \Rightarrow \alpha_i = 1 \Rightarrow Q^TQ = I \Rightarrow Q^T = Q^{-1}$

قضیه: ماتریس متقارن M معین مثبت (نیمه معین) است اگر و فقط اگر هر یک از شرایط زیر برقرار باشد:

۱) همه مقادیر ویژه ماتریس M مثبت (مثبت و صفر) باشند:

اثبات: 1) $\lambda_i \geq 0 : x^TMx = x^TQ\Lambda Q^Tx = \underbrace{x^TQ\Lambda^{\frac{1}{2}}}_{y^T}\underbrace{\Lambda^{\frac{1}{2}}Q^Tx}_{y} = y^Ty \geq 0 \xrightarrow{cause} y \neq 0$

2) $x^TMx \geq 0$; $x^TMx = \underbrace{x^TQ}_{y^T}\Lambda\underbrace{Q^Tx}_{y} = y^T\Lambda y = \sum \lambda_i y_i^2 \geq 0 \Rightarrow each\ \lambda_i \geq 0$

۲) ماتریس غیر ویژه (ویژه) $N_{n\times n}$ وجود داشته باشد بطوری که $M = N^TN$:

اثبات: 1) M , $M = \underbrace{Q\Lambda^{\frac{1}{2}}}_{N^T}\underbrace{\Lambda^{\frac{1}{2}}Q^T}_{N} = N^TN$

با معین مثبت بودن

2) $M = N^TN$; $x^TMx = \underbrace{x^TN^T}_{y^T}\underbrace{Nx}_{y} = y^Ty \geq 0$

۳) دترمینان کهادهای اصلی مقدم (کهادهای اصلی) ماتریس M مثبت (مثبت یا صفر) باشند:

اول کهادهای اصلی و کهادهای اصلی مقدم را تعریف میکنیم:

$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ کهادهای اصلی مقدم: $a_{11}, \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ و کهادهای اصلی: $a_{11}, a_{22}, \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$

$A = \begin{bmatrix} \boxed{a_{11} & a_{12}} & a_{13} \\ a_{21} & \boxed{a_{22} & a_{23}} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$

کهادهای اصلی مقدم: $a_{11}, \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, A$ و کهادهای اصلی: $a_{11}, a_{22}, a_{33}, B, C, \begin{bmatrix} a_{11} & a_{13} \\ a_{31} & a_{33} \end{bmatrix}, A$

← این قضیه میگوید دترمینان کهادهای اصلی مقدم را چک کنیم اگر همه + بودند کافیست اگر نبودند باید دترمینان کهادهای اصلی را چک کنیم اگر همه بزرگتر مساوی صفر بودند (یعنی ماتریس نیمه مثبت است) در غیر این صورت نیست.

Subject :

Year . Month . Date .

Ex. $A = \begin{bmatrix} a & c \\ c & d \end{bmatrix} \xrightarrow{\text{معین مثبت}} \begin{cases} a > 0 \\ ad - c^2 > 0 \end{cases}$

$x^T A x = a x_1^2 + d x_2^2 - 2 c x_1 x_2 \geq a x_1^2 + d x_2^2 - 2|c||x_1||x_2|$

$= (\sqrt{a}|x_1| - \sqrt{d}|x_2|)^2 + (2\sqrt{ad} - 2|c|)|x_1||x_2|$

اگر برقرار باشد این عبارت به ازای هر x_1, x_2 و $x \neq 0$ مثبت باشد باید:

$\sqrt{ad} > |c|$

$\Rightarrow ad - c^2 > 0$

شرط لازم و کافی

Ex. $A = \begin{bmatrix} 10 & 1 & -2 \\ 1 & 4 & -1 \\ -2 & -1 & 1 \end{bmatrix} \rightarrow$ اگر ماتریس متقارن نبود باید $\frac{A+A^T}{2}$ را تشکیل میدادیم و کار را روی آن انجام میدادیم

چون تمام قضایا برای ماتریس متقارن می‌باشد و همچنین اگر مثبت باشد عبارت $x^T A x$ آنگاه $x^T(\frac{A+A^T}{2})x$ هم مثبت است

برای: $10 > 0$, $\begin{vmatrix} 10 & 1 \\ 1 & 4 \end{vmatrix} = 39 > 0$, $|A| = 17 > 0 \Rightarrow$ A معین مثبت می‌باشد.

$eig(A): 0.4, 3.95, 10.63 > 0$ اگر مقادیر ویژه را نیز حساب کردیم،

Ex. $A = \begin{bmatrix} -1 & 1 & -1 \\ 1 & -3 & -2 \\ -1 & -2 & -11 \end{bmatrix} \rightarrow -1 < 0 \Rightarrow$ A معین مثبت نمی‌باشد

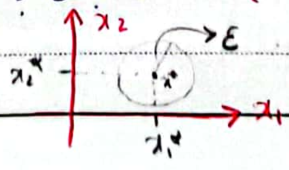
$-A = \begin{bmatrix} 1 & -1 & 1 \\ -1 & 3 & 2 \\ 1 & 2 & 11 \end{bmatrix} \rightarrow 1 > 0, \begin{vmatrix} 1 & -1 \\ -1 & 3 \end{vmatrix} = 2 > 0, |A| > 0 \Rightarrow$ -A معین مثبت است و بنابراین A معین منفی است.

تعاریف بهینه‌سازی: (برای مثال) هدف مینیمم کردن تابع اسکالر است به طوری که تابع، تابعی چند متغیره است (n.):

$f: R^n \rightarrow R \rightarrow \underbrace{D}_{\text{دامنه}} \subseteq R^n, \underbrace{x^*}_{\text{نقطه مینیمم}} \in D$

کمینه محلی (local Min): x^* را روی مجموعه D برای تابع f یک کمینه محلی گویند اگر:

$\exists \varepsilon > 0: \forall x \in D$ with $\|x - x^*\| < \varepsilon : f(x^*) \leq f(x)$



$f(x_1^*, x_2^*) \leq f(x_1, x_2)$

* اگر تعریف فوق به ازای $\forall x \in D$ برقرار باشد، x^* یک global Min میباشد، $f(x^*) \leq f(x)$

* اگر روابط فوق به صورت $f(x^*) < f(x)$ برقرار باشد، در هریک از دو حالت کمینه (global / local)

اکید (Strict) خواهد بود.  غیر اکید، 

نقطه min ما پیدا شده!

شرایط کمینه:
$$f(x^* + \Delta x) = f(x^*) + \left(\frac{\partial f}{\partial x}\Big|_{x=x^*}\right)^T \Delta x + \frac{1}{2}\Delta x^T \left(\frac{\partial^2 f}{\partial x^2}\Big|_{x=x^*}\right)\Delta x + HOT$$

میخواهیم بررسی کنیم در یک همسایگی کوچک از $\|\Delta x\|$ حول x^* (کمینه محلی) باید چه شرایطی برقرار باشد

فرض:
$$\Delta x = -\eta\left(\frac{\partial f}{\partial x}\Big|_{x=x^*}\right) \qquad 0<\eta\ll 1$$

$$\lim_{\eta\to 0}\frac{o(\eta)}{\eta} = 0$$

$$f(x^*+\Delta x) = f(x^*) - \eta\left(\frac{\partial f}{\partial x}\Big|_{x=x^*}\right)^T\left(\frac{\partial f}{\partial x}\Big|_{x=x^*}\right) + o(\eta)$$

$$= f(x^*) - \eta\left\|\frac{\partial f}{\partial x}\Big|_{x=x^*}\right\|^2 + o(\eta)$$

اگر $o(\eta)$ از $-\eta\left\|\frac{\partial f}{\partial x}\right\|^2$ کمتر باشد،
$$-\eta\left\|\frac{\partial f}{\partial x}\Big|_{x=x^*}\right\|^2 + o(\eta) < 0 \Rightarrow f(x^*+\Delta x) < f(x^*)$$
✗

بنابراین اگر x^* یک کمینه محلی برای f باشد و f در x^* مشتق پذیر باشد آنگاه:
$$\frac{\partial f}{\partial x}\Big|_{x=x^*} = 0$$
→ به این شرط لازم مرتبه اول برای کمینه بودن میگویند

شرط لازم مرتبه دوم برای کمینگی:
$$f(x^*+\Delta x) = f(x^*) + 0 + \frac{1}{2}\Delta x^T\frac{\partial^2 f(x^*)}{\partial x^2}\Delta x + o(\|x\|^2)$$

$$\frac{\partial^2 f}{\partial x^2}\Big|_{x=x^*} = H(x^*) \succeq 0 \quad (PSD)$$

اثبات: به برهان خلف میرسیم:
$$H(x^*) < 0 \rightarrow \exists \varepsilon,\ \|\Delta x\| < \varepsilon,\ \frac{1}{2}\Delta x^T H(x^*)\Delta x + o(\|\Delta x\|^2) < 0$$
$$\Rightarrow f(x^*+\Delta x) < f(x^*) \Rightarrow ✗$$

اگر $H(x^*)$ غیر معین هم باشد باز هم به تناقض میرسیم چرا که در آن حالت به ازای برخی راستاها $\frac{1}{2}\Delta x^T H(x^*)\Delta x$ منفی خواهد بود. $\Rightarrow$ ✗

Subject :

Year . Month . Date .

شرط کافی مرتبه دوم : $\left.\frac{\partial^2 f}{\partial x^2}\right|_{x=x^*} = H(x^*) > 0$ (PD)

اثبات : $f(x^*+\Delta x) - f(x^*) = \frac{1}{2}\Delta x^T H(x^*)\Delta x + O(\|x\|^2)$

$$\exists \varepsilon ,\ \|\Delta x\| < \varepsilon \Rightarrow \frac{1}{2}\Delta x^T H(x^*)\Delta x + O(\|x\|^2) > 0 \Rightarrow f(x^*+\Delta x) - f(x^*) > 0$$
$$\Rightarrow f(x^*) < f(x^*+\Delta x)$$

نکته : شرایط مشابه برای تابع $(-f)$ شرایط بیشینه را به دست می‌دهد.

← می‌خواهیم تاثیر مقدارهای ویژه را در اکسترمم بودن بررسی کنیم :

$f(x^*+\Delta x) = f(x^*) + \nabla f(x^*)\Delta x + \Delta x^T H(x^*)\Delta x + HOT \rightarrow$ x^* اکسترمم باشد

Minimum

حال فرض می‌کنیم اثبات شده $\nabla f(x^*) = 0$ می‌باشد، پس :

$f(x^*+\Delta x) = f(x^*) + \Delta x^T H(x^*)\Delta x + HOT$ if $H(x^*) > 0 \rightarrow eig(H(x^*)) > 0$

ماتریس H

اگر بردارهای ویژه v_i ها باشند، چون H full rank می‌باشد، Δx را می‌توان به ترکیب خطی v_i ها نوشت :

مقادیر ویژه

$$\Rightarrow \Delta x^T H(x^*)\Delta x = (\sum \alpha_i v_i)^T H(x^*)(\sum \alpha_i v_i) = (\sum \alpha_i v_i)^T(\sum \alpha_i \lambda_i v_i) = \sum \alpha_i^2 \lambda_i \|v_i\|$$

پس اگر همه λ_i ها مثبت باشند، واضح بود در نتیجه $f(x^*+\Delta x) - f(x^*) > 0$ واضح بود، $f(x^*)$ مینیمم خواهد بود!

نکته : اگر λ_i ای داشته باشیم برابر صفر باشد، آنگاه در Δx نقطه‌ای داریم که در راستای بردار ویژه متناظر با آن λ_i باشد

آنگاه $= \sum \alpha_i \lambda_i \|v_i\|^2 = 0$ خواهد بود و نمی‌توان نتیجه گرفت که $f(x^*)$ مینیمم می‌باشد، پس در $H(x^*) \geq 0$ بودن

شرط لازم است و نه کافی! ولی $H(x^*) > 0$ شرط کافی می‌باشد!

جواب به شکل مجموعه

مثال : نقاط اکسترمم و نوع آن را برای تابع زیر به دست آورید :

$f(x) = x_1^2 + x_2^2 + 3x_1x_2 + 2x_2 - x_1 + 3$

$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$

$$\frac{\partial f}{\partial x} = \begin{pmatrix} 2x_1+3x_2-1 \\ 2x_2+3x_1+2 \end{pmatrix} = 0 \Rightarrow x^* = \begin{pmatrix} -8/5 \\ 7/5 \end{pmatrix}$$

$$H(x) = \frac{\partial^2 f}{\partial x^2} = \begin{pmatrix} 2 & 3 \\ 3 & 2 \end{pmatrix}$$

$\Rightarrow eig(H):\ \det(\lambda I - H) = 0$

$(\lambda-2)^2 - 9 = 0 \Rightarrow \lambda - 2 = \pm 3 \Rightarrow \begin{cases} \lambda = 5 \\ \lambda = -1 \end{cases}$

بنابراین x^* یک نقطه زینی است.

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

شرط لازم مرتبه اول: $\frac{\partial f}{\partial x} = (A+A^T)x + b \xrightarrow[\text{پذیر باشد}]{\text{اگر } A+A^T \text{ معکوس}} x = -(A+A^T)^{-1} b$

$\frac{\partial^2 f}{\partial x^2} = A+A^T$
- PD: کمینه
- ND: بیشینه
- غیر معین، یک سری مقادیر ویژه + و − : زینی

Ex. $f(x) = \frac{1}{2}x_2^2 + x_1 + x_2$

$= x^T \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{2} \end{bmatrix}}_{A} x + \underbrace{(1\ \ 1)}_{b^T} x \Rightarrow A + A^{-1} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$ ⟨ $\lambda = 0$, $\lambda = 1$

$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$

$\frac{\partial f}{\partial x} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} x + \begin{pmatrix} 1 \\ 1 \end{pmatrix} = 0 \Rightarrow \begin{pmatrix} 1 \\ x_2+1 \end{pmatrix} = 0 \Rightarrow 1 = 0 \Rightarrow \times$. ← شرط لازم مرتبه اول برقرار نیست

پس این تابع دارای کمینه نمی‌باشد.

Ex. $f(x) = \frac{1}{2}x_1^2 + \frac{1}{2}x_2^2 + x_1x_2 + x_1 + x_2 = \underline{x}^T \underbrace{\begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix}}_{A} \underline{x} + (1\ \ 1)\underline{x}$

$A + A^T = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \xrightarrow{\text{مقادیر ویژه}} (\lambda-1)^2 - 1 = 0 \Rightarrow \lambda - 1 = \pm 1 \Rightarrow \lambda = 0,\ \lambda = 2$

$\frac{\partial f}{\partial x} = \begin{pmatrix} x_1+x_2+1 \\ x_1+x_2+1 \end{pmatrix} = 0 \Rightarrow x_1 + x_2 + 1 = 0 \Rightarrow$ بی‌نهایت جواب داریم

$f(x) = \frac{1}{2}(x_1+x_2+1)^2 \longrightarrow$ در بی‌نهایت جا صفر می‌شود، پس بی‌نهایت کمینه داریم.

توابع Convex و Concave (محدب و مقعر):

تعریف: تابع f به مجموعه محدب Ω، محدب نامیده می‌شود اگر به ازای هر $x_1, x_2 \in \Omega$ و هر $0 \le \alpha \le 1$ داشته باشیم:

$$f(\alpha x_1 + (1-\alpha)x_2) \le \alpha f(x_1) + (1-\alpha) f(x_2)$$

Subject :

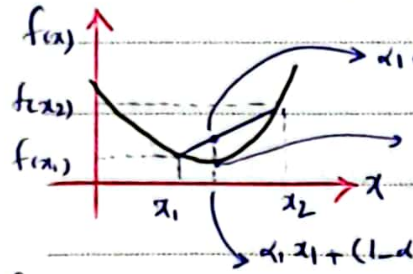
Year . Month . Date .

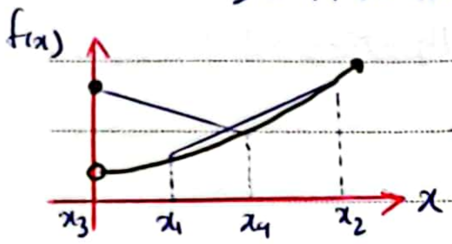
Ω مجموعه محدب : $x_1, x_2 \in \Omega \rightarrow \alpha x_1 + (1-\alpha)x_2 \in \Omega$

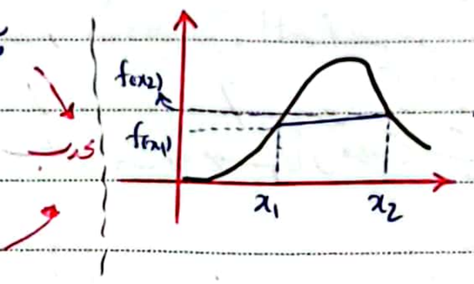
تعریف: اگر به ازای $0<\alpha<1$ و $x_1 \neq x_2$ داشته باشیم:

$$f(\alpha x_1+(1-\alpha)x_2) < \alpha f(x_1)+(1-\alpha)f(x_2)$$

آنگاه f محدب اکید (Strictly Convex) نامیده می‌شود.







تعریف: تابع f بر مجموعه محدب Ω، مقعر (Concave) است اگر $g=-f$ محدب باشد.

تعریف: تابع f بر مجموعه محدب Ω مقعر اکید (Strictly Concave) است اگر $g=-f$ محدب اکید باشد.

خواص تابع Convex:

1) اگر f_1 و ... و f_n توابع محدب و $a_i \geq 0$ باشند آنگاه $\sum_{i=1}^{n} a_i f_i$ محدب است:

اثبات:

$$\times a_1 \quad f_1(\alpha x_1+(1-\alpha_1)x_2) \leq \alpha f_1(x_1)+(1-\alpha_1)f_1(x_2)$$

$$\vdots$$

$$\times a_n \quad f_n(\alpha x_1+(1-\alpha_1)x_2) \leq \alpha f_n(x_1)+(1-\alpha_1)f(x_2)$$

جمع

$$\sum a_i f_i(\alpha x_1+(1-\alpha)x_2) \leq \alpha \sum a_i f_i(x_1) + (1-\alpha)\sum a_i f_i(x_2)$$

2) اگر f تابعی محدب بر مجموعه محدب Ω باشد، آنگاه $\Gamma_c = \{x : x \in \Omega,\ f(x) \leq c\}$ یک مجموعه محدب است:

اثبات:

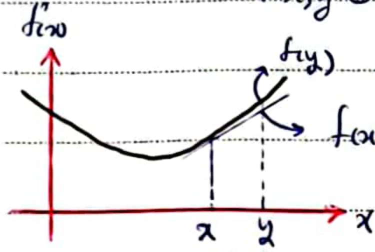
$x_1 \in \Gamma_c \rightarrow f(x_1) \leq c$

$x_2 \in \Gamma_c \rightarrow f(x_2) \leq c$

$\rightarrow f(\alpha x_1+(1-\alpha)x_2) \leq \alpha f(x_1)+(1-\alpha)f(x_2) \leq c$

$\Rightarrow \alpha x_1+(1-\alpha)x_2 \in \Gamma_c \Rightarrow$ ✓

۳) اگر $f \in C^1$ باشد، آنگاه f بر Ω محدب است (محدب اکید) اگر و فقط اگر $f(y) \geq f(x) + \nabla f(x)(y-x)$ $(>)$ $\forall x,y \in \Omega$.



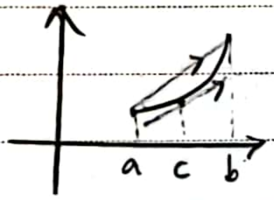
۴) اگر $f \in C^2$ و f تابعی محدب بر مجموعه محدب Ω باشد، آنگاه ماتریس هسین H تابع f نیمه‌معین مثبت بر Ω است.

(در حالت کلی) (اگر ماتریس هسین $H_f > 0$ (معین مثبت) باشد، آنگاه f محدب اکید است اما برعکس آن درست نیست)

اثبات: با استفاده از بسط تیلور - قضیه مقدار میانگین مرتبه دوم داریم

$$f(y) = f(x) + \nabla f(x)(y-x) + \frac{1}{2}(y-x)^T H_f\big(\underline{x+\alpha(y-x)}\big)(y-x)$$

پاره خط بین نقطه‌ای اول و دوم که به ما می‌گوید تساوی داشته باشیم



برای درجه یک داشتیم:

$$f'(c) = \frac{f(b)-f(a)}{b-a}, \quad c \in [a,b]$$

$$f(b) = f(a) + (b-a)f'(c)$$

- اثبات رفت: $H_f \geq 0$: $(y-x)^T H(x+\alpha(y-x))(y-x) \geq 0$
  $\Rightarrow f(y) \geq f(x) + \nabla f(x)(y-x) \Rightarrow f$ محدب است

- اثبات برگشت: برهان خلف: $H_f \geq 0$ نیست، دنبال این:

$\exists z, x,\ z^T H_f(x) z < 0 \rightarrow$ اگر y را طوری انتخاب کنیم: ① در راستای z باشد $y-x$ ② به اندازه کافی کوچک باشد

اگر x و y خیلی نزدیک باشند، با توجه به پیوستگی داریم $\Rightarrow (y-x)^T H_f(x)(y-x) < 0 \rightarrow$

H تابع پیوسته: $H_f(x+\alpha(y-x)) = H_f(x)$ خواهد بود در نتیجه:

$f(y) < f(x) + \nabla f(x)(y-x)$

پس f محدب نخواهد بود!

۵) اگر تابع f بر مجموعه محدب Ω محدب باشد، آنگاه هر کمینه محلی (local) f، کمینه global است:

اثبات: x^*: کمینه محلی برای f $\Rightarrow \exists y\in\Omega : f(y) < f(x^*)$

$$\left.\begin{array}{l} f(\alpha y+(1-\alpha)x^*) \le \alpha f(y)+(1-\alpha)f(x^*) < f(x^*) \\ \{0<\alpha\le 1\} \end{array}\right\} \Rightarrow \dots$$

$$\lim_{\alpha\to 0} \alpha y+(1-\alpha)x^* = x^* \Rightarrow \lim_{\alpha\to 0} f(\alpha y+(1-\alpha)x^*) = f(x^*)$$

$$\dots \Rightarrow f(x^*) < f(x^*) \Rightarrow \times \Rightarrow \dots ✓$$

۶) مجموعه نقاط کمینه تابع f بر مجموعه محدب Ω، $(M\subset\Omega)$ محدب است. $x,y\in M \Rightarrow \alpha x+(1-\alpha)y\in M$

اثبات: $\alpha x+(1-\alpha)y\in\Omega \to f(\alpha x+(1-\alpha)y) \le \alpha f(x)+(1-\alpha)f(y) = f(x)=f(y)$

چون $f(x)$، $f(y)$ کمینه هستند: $f(\alpha x+(1-\alpha)y) = f(x) = f(y)$

$\Rightarrow \alpha x+(1-\alpha)y \in M$

۷) تابع محدب اکید اگر کمینه داشته باشد آن کمینه یکتا است.

اثبات: فرض کنیم کمینه یکتا نباشد: $x,y\in M \Rightarrow \alpha x+(1-\alpha)y\in M$

$\Rightarrow f(x) = f(\alpha x+(1-\alpha)y) < \alpha f(x)+(1-\alpha)f(y) = f(x) \Rightarrow \times$

۸) فرض کنید تابع مشتق‌پذیر و محدب f بر مجموعه محدب Ω تعریف شده است. شرط لازم و کافی برای اینکه $x^*\in\Omega$ یک global Min باشد این است که: $\nabla f(x^*)=0 \rightarrow$ قبلا لازم بودن این شرط ثابت شده بود حالا کافی بودنش رو.

اثبات کافی بودن: $f(y) \ge f(x)+\nabla f(x)(y-x) \xrightarrow[f(x^*)=0]{x=x^*} f(y)\ge f(x^*)\quad \forall y\in\Omega$

این خاصیت برای زمانی است که تابع در نقطه x^* مشتق‌پذیر باشد!

مثال: توابع درجه دو:

$$f(x)=\frac{1}{2}x^T P_{n\times n}\, x + q^T_{1\times n}\, x + r$$

از اول می‌توان فرض کرد $P=P^T$ باشد، اگر P متقارن نبود $\frac{P+P^T}{2}$ جزء متقارن آن است که در ماتریس

باقی ضرایب صفر می‌مانند و این قبلا اثبات شده است.

$$\nabla f^T=\frac{1}{2}(P+P^T)x+q=Px+q=0 \Rightarrow \boxed{Px=-q}$$

$\nabla^2 f = H = P$ اگر $P\succeq 0$ در این صورت f محدب است، اگر $P\succ 0$ در این صورت f محدب اکید است.

در این حالت $x=-P^{-1}q$ مینیمم یکتا است.

در این حالت H، full rank نیست، پس $Px=-q$ یا هیچ جواب یا بی‌شمار جواب دارد.

Ex. $f(x)=x^4 \quad f'(x)=4x^3,\ f''(x)=12x^2=0 \Rightarrow x=0 \Rightarrow H_f\succeq 0$

با اینکه H_f نیمه معین مثبت است، تابع f اکیدا محدب است.

$H_f\succ 0 \rightarrow$ محدب اکید

محدب اکید $\not\rightarrow H_f\succ 0$

Ex. $f(x,y)=x^2/y \quad y>0 \quad \nabla f^T(x,y)=\begin{pmatrix}2x/y\\ -x^2/y^2\end{pmatrix}$

$$\nabla^2 f(x,y)=\begin{pmatrix}2/y & -2x/y^2\\ -2x/y^2 & 2x^2/y^3\end{pmatrix}=H(x,y)=\frac{2}{y^3}\begin{pmatrix}y^2 & -xy\\ -xy & x^2\end{pmatrix}\rightarrow x^2,\ y^2,\ \begin{vmatrix}y^2 & -xy\\ -xy & x^2\end{vmatrix}\geq 0$$

$\Rightarrow$ نیمه معین مثبت می‌باشد

$$\nabla^T f(x,y)=\begin{pmatrix}2x/y\\ -x^2/y^2\end{pmatrix}=0 \rightarrow \begin{cases}x=0\\ y>0\end{cases} \rightarrow (0,y) \rightarrow f(x,y)$$ نقاط کمینه تابع می‌باشند

الگوریتم‌های بهینه‌سازی: ۱) الگوریتم مرتبه اول (تندترین شیب (SD)).

$$f(x+\Delta x) = f(x) + f_x^T \Delta x + \frac{1}{2}\Delta x^T H(x)\Delta x + HOT$$

$$f(\overbrace{x_k+\Delta x_k}^{x_{k+1}}) \simeq f(x_k) + f_x^T(x_k)\Delta x_k \quad \left\{\begin{array}{l} \Delta x_k = x_{k+1} - x_k \\ x_{k+1} = x_k + \Delta x_k \end{array}\right.$$ در جهت dk

اندازه و جهت دارد.

$$\Delta x_k = \vartheta_k u_k \rightarrow u_k = -\frac{f_x(x_k)}{\|f_x(x_k)\|}, \quad \vartheta_k^* = \underset{\vartheta_k}{\text{Argmin}}\, f(\underbrace{x_k + \vartheta_k u_k}_{x_{k+1}})$$

اگر بتوانیم این مسئله را حل کنیم که عالی، وگرنه باید ϑ_k را طوری تعیین کنیم که دست کم برسیم تا از Min عبور نکنیم!

تست خاتمه محاسبات: الگوریتم متوقف می‌شود، در صورتی که یکی از معیارهای زیر برآورده شود:

1) $\|\nabla f(x_k)\| \leq \varepsilon_\nabla$
2) $|f(x_i) - f(x_{i-1})| < \varepsilon_f \rightarrow$ برای چند تکرار پی در پی!
3) $\|x_i - x_{i-1}\| < \varepsilon_x \rightarrow$ " " " !
4) $\vartheta_k \leq k\vartheta_0 \quad 0<k<1$
5) $a =$ Number of iterations

الگوریتم SD برای توابع مربعی: $A>0$ دقت کنید:

$$f(x) = \frac{1}{2}x^T A x + b^T x + c \rightarrow f_x(x) = Ax + b$$

$$\hookrightarrow x_{k+1} = x_k - \vartheta_k\left(\frac{Ax_k+b}{\|Ax_k+b\|}\right)$$

$$f(x_k+\vartheta_k u_k) = \frac{1}{2}(x_k+\vartheta_k u_k)^T A(x_k+\vartheta_k u_k) + b^T(x_k+\vartheta_k u_k) + c$$

$$\frac{\partial f}{\partial \vartheta_k} = \left(\frac{\partial f}{\partial(x_k+\vartheta_k u_k)}\right)^T \frac{\partial(x_k+\vartheta_k u_k)}{\partial x_k} = [A(x_k+\vartheta_k u_k)+b]^T u_k = 0$$

$$\Rightarrow [A(x_k+\vartheta_k u_k)+b]^T u_k = x_k^T A u_k + \vartheta_k u_k^T A u_k + b^T u_k = 0$$

Cause $H_x = u_k^T A u_k > 0 \Rightarrow \vartheta_k = \dfrac{-u_k^T(Ax_k+b)}{u_k^T A u_k} = -\dfrac{u_k^T f_x(x_k)}{u_k^T A u_k}$

$$u_k = \frac{-f_x(u_k)}{\|f_x(u_k)\|} \quad \Bigg\} \Rightarrow \vartheta_k = \frac{f_x^T(x_k) f_x(x_k)}{f_x^T(x_k) A f_x(x_k)} \times \|f_x(x_k)\|$$

$$x_{k+1} = x_k + \Delta x_k = x_k + \alpha_k u_k = x_k - \frac{f_x^T(x_k) f_x(x_k)}{f_x^T(x_k) A f_x(x_k)} f_x(x_k)$$

Ex. $f(x) = x_1^2 + 25x_2^2 = \frac{1}{2} x^T \begin{pmatrix} 2 & 0 \\ 0 & 50 \end{pmatrix} x$, $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$

$f_x(x) = \begin{pmatrix} 5x_1 \\ 50x_2 \end{pmatrix}$, $H = A > 0$

Starting point : $x_0 = \begin{pmatrix} 2 \\ 2 \end{pmatrix}$, $f(x_0) = 104$, $f_x(x_0) = \begin{pmatrix} 4 \\ 100 \end{pmatrix}$

$$\Rightarrow x_1 = x_0 - \frac{(4 \;\; 100)\begin{pmatrix} 4 \\ 100 \end{pmatrix}}{(4 \;\; 100)\begin{pmatrix} 2 & 0 \\ 0 & 50 \end{pmatrix}\begin{pmatrix} 4 \\ 100 \end{pmatrix}} \begin{pmatrix} 4 \\ 100 \end{pmatrix} = \begin{pmatrix} 1.92 \\ -0.003 \end{pmatrix} \rightarrow f(x_1) = 3.64$$

$104 \rightarrow 3.64$ ، رسیدیم

Ex.
$$\begin{cases} 3x_1 - \cos(x_2 x_3) - \frac{3}{2} = 0 \rightarrow e_1 = 3x_1 - \cos(x_2 x_3) - \frac{3}{2} \\ 4x_1^2 - 625 x_2^2 + 2x_2 - 1 = 0 \rightarrow e_2(x) = 4x_1^2 - 625 x_2^2 + 2x_2 - 1 \\ \exp(-x_1 x_2) + 20x_3 - \frac{10\pi - 3}{3} = 0 \rightarrow e_3(x) = \exp(-x_1 x_2) + 20x_3 - \frac{10\pi - 3}{3} \end{cases} \quad x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

$F(x) = \frac{1}{2}\left(e_1^2(x) + e_2^2(x) + e_3^2(x)\right) \rightarrow$ اگر این را مینیمم کنیم، به x مورد نظر می‌رسیم

$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \rightarrow F(x_0) = \frac{1}{2}\left((-2.5)^2 + (-1)^2 + (10.472)^2\right) = 58.456$, $F_x(x_0) = \begin{pmatrix} -7.5 \\ -2 \\ 209.44 \end{pmatrix}$

$x_1 = x_0 - \alpha_0 \frac{F_x(x_0)}{\|F_x(x_0)\|} = x_0 - \alpha_0 \begin{pmatrix} -7.5 \\ -2 \\ 209.44 \end{pmatrix}$

در اینجا بهینه سازی نشده، صرفاً کافی است که برسیم به $F(x_1) < F(x_0)$ ، برسد

$\alpha_0 = 0.001 \Rightarrow x_1 = \begin{pmatrix} 0.0075 \\ 0.002 \\ -0.20944 \end{pmatrix} \Rightarrow F(x_1) = 23.306 < 58.456$

روش SD : مزیت ، سریع به جواب بهینه نزدیک می‌شود . اشکال : وقتی نزدیک می‌رسد ، از تقریب شدن به جواب بهینه

روش مرتبه دوم نیوتن :

$$f(x_k + \Delta x_k) \simeq f(x_k) + \nabla f(x_k)^T \Delta x_k + \frac{1}{2} \Delta x_k^T H(x_k) \Delta x_k$$

$$\frac{\partial f(x_k + \Delta x_k)}{\partial \Delta x_k} = \nabla^T f(x_k) + H(x_k) \Delta x_k = 0 \xrightarrow{\text{فرض : } H(x_k) > 0} \Delta x_k = -H^{-1}(x_k) \nabla^T f(x_k)$$

$$\Rightarrow x_{k+1} = x_k - \underbrace{H^{-1}(x_k)}\nabla^T f(x_k)$$

برای محاسبه معکوس H_k ، هزینه زیادی باید پرداخت شود

Subject :

Year . Month . Date .

Ex. $f(x) = \frac{1}{2}x^TAx + b^Tx + c \rightarrow \nabla^T f(x) = Ax + b \quad , \quad H(x) = A > 0$

شرایط بهینگی

$x_1 = x_0 - A^{-1}(Ax_0 + b) = x_0 - x_0 - A^{-1}b = -A^{-1}b$

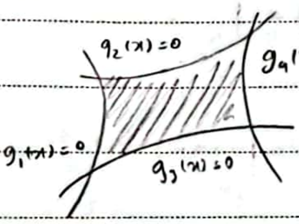
$\nabla^T f(x) = Ax + b = 0 \Rightarrow x^* = -A^{-1}b$

بنابراین برای تابع مربعی الگوریتم نیوتن تنها یک مرحله به مقدار کمینه تابع می‌رسد.

- یک ایده برای استفاده از مزایای روش SD و نیوتن: استفاده از SD در مراحل اولیه و نزدیک شدن به جواب بهینه و سپس استفاده از روش نیوتن جهت تسریع در همگرایی به نقطه بهینه.

بهینه‌سازی مقید:

$$\text{Minimize } f(x) \qquad x \in \Omega \subset R^n \qquad \begin{cases} h_1(x) = 0 & g_1(x) \le 0 \\ h_2(x) = 0 & g_2(x) \le 0 \\ \vdots & \vdots \\ h_m(x) = 0 & g_p(x) \le 0 \end{cases} \qquad m \le n$$



فرض: f و g_i و h_j ها پیوسته و مشتق‌پذیر از مرتبه دوم هستند.

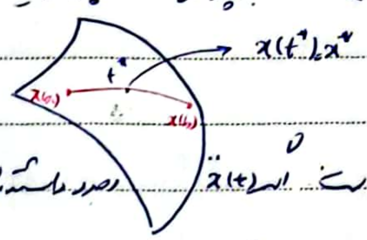
$j = 1, \ldots, m \qquad i = 1, \ldots, p$

- تعریف: به مجموعه‌ای از x ها که در همه قیود صدق کنند feasible یا شدنی می‌گویند.

قیود تساوی: با فرض استقلال قیود، تساوی مجموعه قیود یک ابرصفحه با بعد $n-m$ را تشکیل می‌دهند.

- صفحه (ابرصفحه) مشخص شده با قیود را با S نمایش می‌دهیم.

- یک مسیر بر صفحه S را با مجموعه نقاط $x(t) \in S$ نشان می‌دهیم که در بازه $a \le t \le b$ با پارامتر t، پارامتریزه شده است.



- یک مسیر مشتق‌پذیر است اگر $\dot{x} = \frac{dx}{dt}$ وجود داشته باشد و مشتق‌پذیر از مرتبه دوم است اگر $\ddot{x}(t)$ وجود داشته باشد.

- مسیر $x(t)$ از نقطه x^* بر S عبور می‌کند اگر وجود داشته باشد $t^* \in \{t : a \le t \le b\}$ به طوری که $x(t^*) = x^*$

- صفحه مماس: اگر همه مسیرهای مشتق‌پذیر از S که از x^* عبور می‌کنند را در نظر بگیریم؛ صفحه مماس در x^* به مجموعه مشتق‌های همه مسیرها در x^* گفته می‌شود.

تعریف: به نقطه x^* که $h(x^*)=0$ را برآورده می‌کند، رگولار گویند اگر بردارهای گرادیان $\nabla h_1(x^*), \dots, \nabla h_m(x^*)$ مستقل خطی باشند.

بهینه‌سازی مقید - قیود تساوی:

قضیه: در نقطه رگولار x^* بر صفحه که با $h(x^*)=0$ مشخص شده است صفحه مماس به صورت زیر تعریف می‌گردد:

$$M=\{y : \nabla h(x^*)\,y=0\}$$

اثبات: $x(t)$ مسیری بر صفحه که می‌باشد که از x^* می‌گذرد.

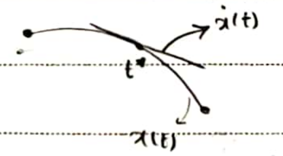
$h(x(t))=0$

مشتق نسبت به t

$$\underbrace{\nabla h(x(t))}_{m\times n}\,\dot{x}(t)\Big|_{t=t^*}=0$$

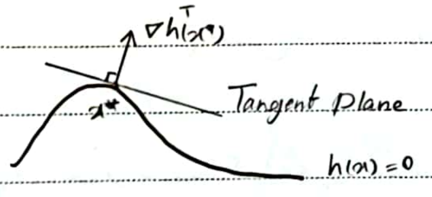
* h در واقع ماتریسی از تمام قیود تساوی $\left\{\begin{array}{l} h_1=0\\ \vdots \\ h_m=0\end{array}\right.$ می‌باشد.

$$\nabla h(x)=\begin{bmatrix}\nabla h_1(x)\\ \nabla h_2(x)\\ \vdots\\ \nabla h_m(x)\end{bmatrix}_{m\times n}$$

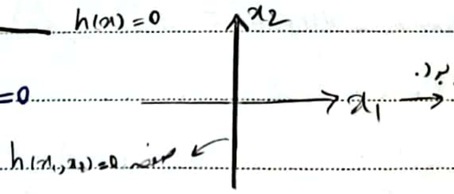


بنابراین بردار گرادیان بر همه‌ی مشتق‌های مسیر گذرنده از x^* عمود است، پس بر کل صفحه‌ی مماس در x^* عمود است.

• بنابراین می‌توان گفت که صفحه‌ی مماس آن صفحه‌ای است که $\nabla h(x^*)$ بر آن عمود باشد! این نتیجه‌گیری را می‌توان با توجه به اینکه ∇h_iها مستقل هستند انجام داد. در واقع ماتریس $\nabla h(x)$ full rank می‌باشد $(\text{rank}=m)$ و فضای پوچی آن $n-m$ بعدی می‌باشد که $\dot{x}(t)$ در فضای پوچی می‌باشد!



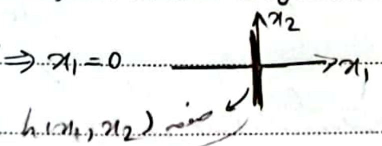
Ex. $h(x_1,x_2)=x_1=0$



$\longrightarrow$ صفحه‌ی مماس بر نقطه دلخواه $x^*=(0,x_2^*)$ محور x_2 خواهد بود.

راه ریاضی: $\nabla h(x_1,x_2)=(1\ \ 0) \rightarrow (1\ \ 0)\,y=(1\ \ 0)\begin{pmatrix}y_1\\ y_2\end{pmatrix}=0 \Rightarrow y_1=0 \Rightarrow M=\{y : y_1=0\}$

Ex. $h(x_1,x_2)=x_1^2=0 \Rightarrow x_1=0$



$\nabla h(x^*)=(2x_1^*\ \ 0) \xrightarrow{x_1=0} =(0,0) \Rightarrow$ بردار گرادیان مستقل نیست، x^* رگولار نیست!

$\forall y\in R^2,\ \nabla h(x^*)\,y=0 \Rightarrow M=R^2 \Rightarrow$ صفحه مماس

×

Subject :

Year . Month . Date .

لم: اگر x^* نقطه‌ی regular قید $h(x)=0$ باشد و همچنین نقطه‌ی اکسترمم محلی برای f با توجه به قیود باشد، آنگاه به ازای هر $y \in \mathbb{R}^n$

اگر $\nabla h(x^*)y = 0$ آنگاه $\nabla f(x^*)y = 0$ خواهد بود. با استفاده از شرایط گفته شده برقرار باشد، این نکته: گرادیان f هم بر صفحه‌ی مماس عمود است.

اثبات: با این نکته توجه می‌کنیم که x^* یک اکسترمم محلی برای f می‌باشد. اگر مسیر $x(t)$ از x^* عبور کند، می‌دانیم:

$$\frac{d}{dt} f(x(t))\Big|_{t=t^*} = 0 \rightarrow \nabla f(x(t^*))\dot{x}(t^*) = 0 \Rightarrow$$

بردار گرادیان f بر بردار مشتق مسیر $x(t)$ در نقطه‌ی t^* (یا همان x^*) عمود می‌باشد.

چون $x(t)$ یک مسیر دلخواه بود، بنابراین بردار گرادیان f بر تمام بردارهای مشتق مسیرها که از t^* (یا x^*) می‌گذرند عمود می‌باشد.

بنابراین $\nabla f(x^*)$ بر همه بردارهای y که در عبارت $\nabla h(x^*)y = 0$ صدق می‌کنند (یعنی همان صفحه مماس) عمود می‌باشد در x^*.

نتیجه‌گیری که از لم بالا گرفته می‌شود به شرح زیر است: صفحه‌ی مماس $n-m$ بعدی می‌باشد و صفحه‌ی گفته شده $\{\nabla h_1, \dots, \nabla h_m\}$

بر این صفحه عمود می‌باشد که در واقع فضای Null آن را تشکیل می‌دهند و دارای بعد m می‌باشند (به علت مستقل بودن ∇h_iها).

∇h_iها تمام فضای Null را پوشش می‌دهند. با حال گفته شد که $\nabla f(x^*)$ هم بر صفحه‌ی مماس عمود می‌باشد! پس بردار گرادیان

f نسبت به $\{\nabla h_1(x^*), \dots, \nabla h_m(x^*)\}$ وابسته است، یعنی با یک ترکیب خطی از ∇h_iها می‌توان به $\nabla f(x^*)$ رسید.

قضیه: اگر x^* یک کمینه محلی f با توجه به قید $h(x)=0$ باشد و x^* یک نقطه regular برای این قیود باشد، آنگاه

یک بردار $\lambda \in \mathbb{R}^m$ وجود دارد به طوری که: $\nabla f(x^*) + \lambda^T \nabla h(x^*) = 0$ ← در واقع همان نتیجه‌گیری بالا می‌باشد!

اثبات:

$$\underset{1\times n}{[\nabla f(x^*)]} + [\lambda_1 \dots \lambda_m]\underset{m\times n}{\begin{bmatrix} \nabla h_1(x^*) \\ \vdots \\ \nabla h_m(x^*) \end{bmatrix}} = 0 = [\nabla f(x^*)] + \sum_{i=1}^{m} \lambda_i \nabla h_i(x^*) = 0$$

← می‌توان شرط لازم بهینگی را نسبت به تابع جدید لاگرانژین گفته شده بیان کنیم: $l(x,\lambda) = f(x) + \lambda^T h(x)$

$$\begin{array}{l} \frac{\partial l}{\partial x} = 0 \rightarrow \\ \frac{\partial l}{\partial \lambda} = 0 \rightarrow \end{array} \begin{cases} \nabla f(x) + \lambda^T \nabla h(x) = 0 \rightarrow n \text{ معادله} \\ h(x) = 0 \rightarrow m \text{ معادله} \end{cases} \Rightarrow n+m \text{ معادله و } \quad \begin{array}{l} \lambda = \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{pmatrix} \\ x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \end{array} \Big\} \rightarrow n+m \text{ مجهول}$$

• فرض کنید به جای این شرط‌ها، دستگاه معادلات حل شود و بدست آوریم، اما این است که اگر بخواهیم با SD این مسئله را حل کنیم، نقطه‌ی شروع باید در

Batus دامنه همگرایی باشد و در این صورت نقطه‌ی بهینه را پیدا می‌کنیم!

Subject :

Year .   Month .   Date .

Ex. Maximize $x_1x_2 + x_2x_3 + x_1x_3$ s.t $x_1 + x_2 + x_3 = 3$

$\equiv$ Minimize $\underbrace{-x_1x_2 - x_2x_3 - x_1x_3}_{f(x)}$ $x_1 + x_2 + x_3 = 3$

$$\left.\begin{cases} \nabla f(x) + \lambda^T \nabla h(x) = -(x_2+x_3,\ x_1+x_3,\ x_1+x_2) + \lambda(1\ \ 1\ \ 1) = 0 \\ h(x) = 0 \Rightarrow x_2 + x_3 + x_1 = 3 \end{cases}\right\} \Rightarrow \lambda = 2,\ x_1 = x_2 = x_3 = 1$$

Ex. Maximize xyz s.t $xy + yz + xz = \frac{c}{2}$ $c > 0$

$$\nabla f(x) + \lambda^T \nabla h(x) = \begin{array}{c} ① \\ ② \\ ③ \end{array}\begin{pmatrix} -yz \\ -xz \\ -xy \end{pmatrix} + \lambda \begin{pmatrix} y+z \\ x+z \\ x+y \end{pmatrix} = 0$$

$h(x) = 0 \rightarrow xy + yz + xz = \frac{c}{2}$

$$\left.\begin{aligned} &x \times ① - y \times ② = \lambda(x-y)z = 0 \\ &y \times ② - z \times ③ = \lambda(y-z)x = 0 \\ &z \times ③ - x \times ① = \lambda(z-x)y = 0 \end{aligned}\right\} (*)$$

$$\left.\begin{aligned} &① + ② + ③ = -\underbrace{(xy + yz + xz)}_{\frac{c}{2}} + 2\lambda(x+y+z) = 0 \\ &\Rightarrow 2\lambda(x+y+z) = \frac{c}{2} \end{aligned}\right\} (**)$$

می‌خواهیم ببینیم که آیا λ و x و y و z می‌توانند صفر باشند یا خیر!

(I) $2\lambda(x+y+z) = \frac{c}{2} \longrightarrow$ if $\lambda = 0 \Rightarrow 0 = \frac{c}{2} > 0 \Rightarrow$ ✗

(II) $x = 0 \Rightarrow \lambda z = 0 \Rightarrow z = 0 \Rightarrow \lambda y = 0 \Rightarrow y = 0 \longrightarrow$ به طریق مشابه برای حالات $z=0$ و $y=0$ به تناقض می‌رسیم که تمام متغیرها باید صفر باشند!

بنابراین $x \neq 0$ و $y \neq 0$ و $z \neq 0$ و $\lambda \neq 0$

$$\left.\begin{aligned} &(*) \\ &(**) \end{aligned}\right\} \begin{aligned} &x = y = z = \pm\sqrt{\frac{c}{6}} \\ &\lambda = \pm\frac{1}{2}\sqrt{\frac{c}{6}} \end{aligned}$$

شرایط لازم و کافی مرتبه دوم:

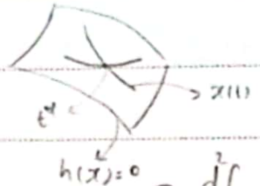
۱) شرایط لازم: اگر x^* بهینه نسبی f مقید به $h(x) = 0$ و رگولار نسبت به قیود باشد، $M = \{y : \nabla h(x^*) y = 0\}$ صفحه مماس بر x^* باشد، آنگاه ماتریس $L(x^*)$ (ماتریس هسین لاگرانژین) بر M نیمه معین مثبت است.

$$L(x^*) = \nabla^2 f(x^*) + \lambda^T \nabla^2 h(x^*) = F(x^*) + \lambda^T H(x^*) \rightarrow \forall y \in M,\ y^T L(x^*) y \geq 0$$

اثبات در صفحه بعد.

Subject :

Year . Month . Date .



we know: $\left.\frac{d^2f}{dt^2}\right|_{t=t^*} \geq 0$

① $\frac{d^2f}{dt^2} = \frac{d}{dt}\left(\nabla f(x(t))\dot{x}(t)\right) = \dot{x}^T(t)\nabla^2 f(x(t))\dot{x}(t) + \nabla f(x(t))\ddot{x}(t) \geq 0$

$$\lambda^T h(x)=0 \xrightarrow{\partial/\partial t} \lambda^T\nabla h(x(t))\dot{x}(t) = \sum \lambda_i \nabla h_i(x(t))\dot{x}(t) = 0$$

$$\xrightarrow{\partial^2/\partial t^2} = \dot{x}^T(t)\underbrace{\sum_{i=1}^{m}\lambda_i\nabla^2 h_i(x(t))}_{\lambda^T\nabla^2 h(x(t))}\dot{x}(t) + \sum_{i=1}^{m}\lambda_i\nabla h_i(x(t))\ddot{x}(t) = 0$$

② $\dot{x}^T(t)\lambda^T\nabla^2 h(x(t))\dot{x}(t) + \lambda^T\nabla h(x(t))\ddot{x}(t) = 0$

$$\rightarrow ①+② \geq 0 \Rightarrow \dot{x}(t)\left[\nabla^2 f(x(t)) + \lambda^T\nabla^2 h(x(t))\right]\dot{x}(t) + \left.\underbrace{\left[\nabla f(x(t)) + \lambda^T\nabla h(x(t))\right]}_{\text{at } t=t^* \ =0}\ddot{x}(t)\right|_{t=t^*} \geq 0$$

$$\Rightarrow \dot{x}(t^*)\left[\nabla^2 f(x^*) + \lambda^T\nabla^2 h(x^*)\right]\dot{x}(t^*) \geq 0$$

$$= \dot{x}(t^*)\left[F(x^*) + \lambda^T H(x^*)\right]\dot{x}(t^*) \geq 0 \Rightarrow y^T\left[F(x^*) + \lambda^T H(x^*)\right]y \geq 0 \ \text{for} \ \forall y \in M$$

که به ازای هر مسیر دلخواهی از t^*

$\Rightarrow$ For all $y \in M$: $y^T L(x^*) y \geq 0$

شرط کافی مرتبه دوم: اگر x^* یک نقطه رگولار برای $h(x)=0$ باشد به طوری که $h(x^*)=0$ را برقرار داشته باشد

و $\lambda \in R^m$ به طوری که $\nabla f(x^*) + \lambda^T\nabla h(x^*) = 0$، آنگاه اگر $L(x^*)$ بر $M=\{y : \nabla h(x^*)y=0\}$ معین مثبت باشد

$\rightarrow y^T L(x^*) y > 0, \ \forall y \in M$

آنگاه x^* یک نقطه مینیمم محلی اکید برای تابع f مقید به $h(x)=0$ می‌باشد.

Ex. Max $x_1x_2 + x_2x_3 + x_1x_3$ s.t. $x_1+x_2+x_3=3$

Min $-x_1x_2 - x_2x_3 - x_1x_3 = f(x) \rightarrow \nabla^T f = -\begin{pmatrix} x_2+x_3 \\ x_1+x_3 \\ x_1+x_2 \end{pmatrix}$, $\nabla^2 f = -\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$

$h(x) = x_1+x_2+x_3-3=0 \rightarrow \nabla^T h = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$, $\nabla^2 h = 0$

$\nabla^2 f(x^*) + \lambda^T\nabla^2 h = -\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \longrightarrow$ با بررسی کهادهای اصلی و دترمینان‌های اصلی مقدم متوجه می‌شویم که نه PSD است و نه

NSD! (دترمینال ۲ منفی می‌شه!)

که شرط کافی مرتبه دوم می‌گه ماتریس $-\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$ باید برای $y\in M$ ها PSD باشد نه برای همه نقاط!

حالا به بررسی حالت کلی مسئله می‌پردازیم:

Batus

Subject :

Year . Month . Date .

به صورت زیر:

$$\left\{\begin{array}{l} \underset{1\times n}{\nabla f(x^*)} + \underset{1\times m}{\lambda^T} \underset{m\times n}{\nabla h(x^*)} + \underset{1\times n}{\mu^T} \underset{p\times n}{\nabla g(x^*)} = 0 \\ \mu^T g(x^*) = 0 \rightarrow \sum_j \mu_j g_j(x^*) = 0 \\ h(x^*) = 0 \end{array}\right.$$

چون x^* کمینه نسبی تابع f با توجه به قیدهای موجود است، همان نقطه کمینه نسبی f با توجه به قیدهای فعال می‌باشد، بنابراین برای قیدهای فعال

همان رابطه لاگرانژ برقرار است. ($\mu_j \ge 0$ برای قیدهای فعال g_j در J ($j \in J$) خواهد بود).

برای سایر قیدهای غیرفعال (که نباید در رابطه لاگرانژ ظاهر شوند) باید داشته باشیم $\mu_j = 0$ ($j \notin J$)، بنابراین روی هم رفته (۲):

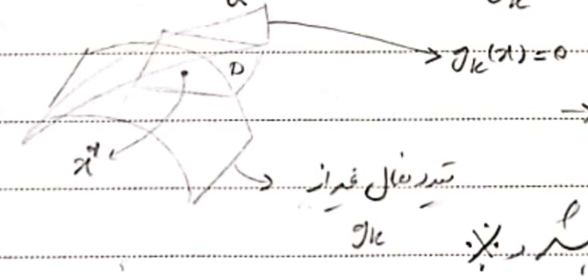
(۲) $\left\{\begin{array}{l} g_j(x^*) < 0 \rightarrow \mu_j = 0 \\ g_j(x^*) = 0 \rightarrow \mu_j \ge 0 \end{array}\right.$

که رابطه $\mu^T g(x^*) = 0$ این خاصیت را برآورده می‌کند ($\mu \ge 0$).

اثبات $\mu \ge 0$: فرض کنید $g_k(x^*) = 0$ و $\mu_k < 0$ باشد و D صفحه مماس حاصل از سایر قیدهای فعال در x^* باشد:

$D: \{y : \nabla h(x^*) y = 0, \nabla g_j(x^*) y = 0, j \in J, j \ne k\}$

آنگاه: $\exists y \in D : x = x^* + \alpha y \quad 0 < \alpha \ll 1 : g_k(x) < 0$ (علت)



(علت): زیرا ابعاد فضای D یک بعد بیشتر است، به سمت جهت آن از x^* $\rightarrow$

حرکت کنیم $g_k(x) < 0$ شود، یعنی $g_k(x)$ در x^* بر D مماس نباشد.

$\Rightarrow g_k(x^* + \alpha y) = \underbrace{g_k(x^*)}_{\text{صفر}} + \nabla g_k(x^*)(\alpha y) + o(\alpha) < 0$

چون به صفر میل می‌کند سریعتر از مرتبه یک

برای برقراری این نامساوی باید $\nabla g_k(x^*) y$ منفی باشد.

$\Rightarrow$ به دست آوردیم: $\nabla g_k(x^*) y < 0$

داشتیم: $\nabla f(x^*) + \lambda^T \nabla h(x^*) + \mu_{-k}^T \nabla g_{-k}(x^*) + \mu_k \nabla g_k(x^*) = 0 \xrightarrow{\text{از سمت راست در y ضرب می‌کنیم}}$

$\longrightarrow \nabla f(x^*) y + \lambda^T \underbrace{\nabla h(x^*) y}_{0} + \mu_{-k}^T \underbrace{\nabla g_{-k}(x^*) y}_{0} + \underbrace{\mu_k}_{<0} \underbrace{\nabla g_k(x^*) y}_{<0} = 0 \Rightarrow \nabla f(x^*) y < 0$

$\xrightarrow{\text{با استفاده از بسط تیلور}} f(x^* + \alpha y) < f(x^*) \Rightarrow$ تناقض با اینکه x^* کمینه نسبی با توجه به قیود است $\Rightarrow \times \Rightarrow \mu_k \ge 0$!

Subject :
Year . Month . Date .

Ex. $\text{Min} \; 2x_1^2 + 2x_1x_2 + x_2^2 - 10x_1 - 10x_2 \quad s.t \quad x_1^2 + x_2^2 \leq 5$, $\quad 3x_1 + x_2 \leq 6 \qquad \nabla g = \begin{pmatrix} \nabla g_1 \\ \nabla g_2 \end{pmatrix}$

$$\nabla f(x^*) + \lambda^T \nabla h(x^*) + \mu^T \nabla g(x^*) = (4x_1 + 2x_2 - 10 \quad 2x_1 + 2x_2 - 10) + (\mu_1 \;\; \mu_2) \begin{pmatrix} 2x_1 & 2x_1 \\ 3 & 1 \end{pmatrix} = 0$$

$$\begin{cases} 4x_1 + 2x_2 - 10 + 2\mu_1 x_1 + 3\mu_2 = 0 \\ 2x_1 + 2x_2 - 10 + 2\mu_1 x_2 + \mu_2 = 0 \\ \mu_1 (x_1^2 + x_2^2 - 5) = 0 \quad \mu_1 \geq 0 \\ \mu_2 (3x_1 + x_2 - 6) = 0 \quad \mu_2 \geq 0 \end{cases}$$

$\Big\} \rightarrow \mu_1 (x_1^2 + x_2^2 - 5) + \mu_2 (3x_1 + x_2 - 6) = 0 \equiv \mu^T g(x) = 0$

به ۴ حالت تقسیم می‌شود: ۱) هیچ‌یک از قیدها فعال نباشد،

$$\mu_1 = 0 = \mu_2 \Rightarrow \begin{rcases} 4x_1 + 2x_2 - 10 = 0 \\ 2x_1 + 2x_2 - 10 = 0 \end{rcases} \rightarrow \begin{matrix} x_1 = 0 \\ x_2 = 5 \end{matrix} \Rightarrow g_1(0,5) = 25 - 5 = 20 \nleq 0 \Rightarrow \times.$$

۲) قید اول فعال باشد و قید دوم فعال نباشد،

$$\mu_2 = 0 \Rightarrow \begin{cases} 4x_1 + 2x_2 - 10 + 2\mu_1 x_1 = 0 \\ 2x_1 + 2x_2 - 10 + 2\mu_1 x_2 = 0 \\ x_1^2 + x_2^2 - 5 = 0 \end{cases} \Rightarrow \begin{cases} x_1 = 1 \\ x_2 = 2 \\ \mu_1 = 1 \geq 0 \; \text{--} \checkmark \end{cases}$$

۳) قید دوم فعال باشد و قید اول فعال نباشد،

$$\mu_1 = 0 \Rightarrow \begin{cases} 4x_1 + 2x_2 - 10 + 3\mu_2 = 0 \\ 2x_1 + 2x_2 - 10 + \mu_2 = 0 \\ 3x_1 + x_2 = 6 \end{cases} \Rightarrow \begin{cases} x_1 = 2/5 \\ x_2 = 6 - 6/5 \\ \mu_2 = -2/5 < 0 \Rightarrow \times. \end{cases}$$

۴) هر دو قید فعال باشند،

$$\begin{cases} 4x_1 + 2x_2 - 10 + 2\mu_1 x_1 + 3\mu_2 = 0 \\ 2x_1 + 2x_2 - 10 + 2\mu_1 x_2 + \mu_2 = 0 \\ x_1^2 + x_2^2 - 5 = 0 \\ 3x_1 + x_2 - 6 = 0 \end{cases} \longrightarrow$$ این دستگاه جواب ندارد!

شرط لازم مرتبه دوم: اگر توابع $h, g, f \in C^2$ باشند و x^* نقطه منظم قیود باشد (قیود فعال) و همچنین x^* یک کمینه نسبی f مقید بر قیود باشد، آنگاه ضرایب $\mu \geq 0$ و $\mu \in R^p$ و $\lambda \in R^m$ وجود دارند به طوری که $L(x^*)$ بر صفحه مماس مربوط به قیود فعال در x^* نیمه معین مثبت (psd) است.

$$L(x^*) = \underbrace{\nabla^2 f(x^*)}_{n \times n} + [\lambda^T \underbrace{\nabla^2 h(x^*)}_{n \times n}] + \mu^T \underbrace{[\nabla^2 g(x^*)]}_{n \times n}$$

$M' = \{ y : \nabla h(x^*) y = 0, \; \nabla g_j(x^*) y = 0 \;\; \forall j \in J \} \longrightarrow$ مجموعه قیدهای فعال!

$\forall y \in M' : \; y^T L(x^*) y \geq 0$

شرایط کافی مرتبه دوم: اگر توابع $f, g, h \in C^2$ و x^* نقطه منظم قید باشد، شرط کافی برای اینکه x^* نقطه بهینه نسبی اکید برای مسئله بهینه‌سازی مقید (مساوی و نامساوی) باشد، این است که وجود داشته باشد $\lambda \in R^m$ و $\mu \in R^p$ به طوری که:

$$\begin{cases} \mu \geq 0 \\ \mu^T g(x^*) = 0 \\ \nabla f(x^*) + \lambda^T \nabla h(x^*) + \mu^T \nabla g(x^*) = 0 \end{cases}$$

و $L(x^*)$ بر M' معین مثبت (PD) باشد یا به عبارتی:
$\forall y \in M' : y^T L(x^*) y > 0$

مثال: مثال قبلی، حالت دوم را در بررسی کافی بررسی می‌کنیم:

$$L(x^*) = \nabla^2 f(x^*) + \lambda^T \nabla^2 h(x^*) + \mu^T \nabla^2 g(x^*) = \begin{pmatrix} 4 & 2 \\ 2 & 2 \end{pmatrix} + (1 \quad 0) \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix} = \begin{pmatrix} 6 & 2 \\ 2 & 4 \end{pmatrix}$$

با چک کردن کهادها: $6>0$ و $\det = 20 > 0$ پس $L(x^*) > 0$ می‌باشد، پس شرط کافی برقرار می‌باشد.

Ex. $\text{Min } \frac{1}{2}(x_1^2 + x_2^2)$

s.t. $x_1 + x_2 = 2 \rightarrow x_1 + x_2 - 2 \leq 0$

$\underbrace{x_1 \geq 0}_{-x_1 \leq 0}$, $\underbrace{x_2 \geq 0}_{-x_2 \leq 0}$

$$\begin{cases} \nabla f(x^*) + \lambda^T \nabla h(x^*) + \mu^T \nabla g(x^*) = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \lambda \begin{pmatrix} 1 \\ 1 \end{pmatrix} + (\mu_1, \mu_2) \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} \quad (\mathrm{I}) \\ \mu_1(-x_1) = 0 \qquad \mu_1 \geq 0 \\ \mu_2(-x_2) = 0 \qquad \mu_2 \geq 0 \end{cases}$$

(I): $x_1 + \lambda - \mu_1 = 0 \qquad -\mu_1 x_1 = 0$

$x_2 + \lambda - \mu_2 = 0 \qquad -\mu_2 x_2 = 0$

$x_1 + x_2 - 2 = 0$

حالت‌بندی می‌کنیم: 1) هر دو قید نامساوی غیرفعال:

$$\mu_1 = \mu_2 = 0 \rightarrow \begin{cases} x_1 + \lambda = 0 \\ x_2 + \lambda = 0 \\ x_1 + x_2 = 0 \end{cases} \rightarrow \begin{array}{l} \lambda = -1 \\ x_1 = x_2 = 1 \end{array}$$

چک کردن شرط کافی: $L(x^*) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + (-1)\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} > 0$

2) قید اول فعال باشد: $x_1 = 0 \Rightarrow \mu_2 = 0 \Rightarrow x_2 = 2 \Rightarrow \lambda = -2 \Rightarrow \mu_1 = -2$ ✗

3) " " 2 " ": $x_2 = 0 \Rightarrow \mu_1 = 0 \Rightarrow x_1 = 2 \Rightarrow \lambda = -2 \Rightarrow \mu_2 = -2$ ✗

4) هر دو شرط فعال: $x_1 = x_2 = 0 \Rightarrow x_1 + x_2 - 2 = -2 = 0$ ✗

Subject :

Year . Month . Date .

حال می‌خواهیم این نکات را برای توابع محدب بررسی کنیم:

نکته: در بهینه‌سازی مقید اگر مجموعه مشخص شده توسط قیود ساده و نامساوی را Ω بنامیم، اگر مجموعه Ω محدب باشد

و همچنین تابع $f(x)$ محدب باشد، ویژگی‌های بیان شده در بهینه‌سازی محدب، در حالت حضور قیود ساده و نامساوی

نیز برقرار می‌باشد.

$$\text{Min } f(x) \quad s.t \quad \left.\begin{array}{ll} g_j(x) \leq 0 & j = 1, \dots, p \\ h_i(x) = 0 & i = 1, \dots, m \end{array}\right\} \Omega$$

حال می‌خواهیم بررسی کنیم به چه صورت Ω محدب می‌شود؟

این مسئله را با ذکر این نکته بررسی می‌کنیم: فضای حالت که از اشتراک دامنه‌های تمام قیدها

نکته: اشتراک چند مجموعه محدب، یک مجموعه محدب است. بنابراین برای اینکه Ω محدب باشد باید تمامی g_jها و

h_iها محدب باشند؛ برای g_jها داریم: (تعمیم زیر را)

تعمیم: اگر Γ برای g_j برقرار باشد، آنگاه g_j محدب می‌باشد:
$\Gamma = \{x, g(x) \leq \beta\}$

برای h_iها روند ساده‌ای را در نظر می‌گیریم؛ در روابع $0 = h_i(x)$ $\left|\begin{array}{l} h_i(x) \leq 0 \\ -h_i(x) \leq 0 \end{array}\right.$ می‌باشد، حال h_i شبیه g_j شد!

بنابراین: $\left\{\begin{array}{l} h_i(x) \text{ باید محدب باشد} \\ -h_i(x) \text{ " " " یعنی } h_i(x) \text{ مقعر باشد} \end{array}\right.$ $\Rightarrow$ $h_i(x)$ هم باید محدب و هم باید مقعر باشد!

همانطور که گفته شد یک تابع (h) در صورتی محدب است که $\nabla^2 h(x) \geq 0$ باشد پس:

$$\left.\begin{array}{l} \nabla^2 h(x) \geq 0 \\ -\nabla^2 h(x) \geq 0 \end{array}\right\} \Rightarrow \nabla^2 h(x) = 0 \Rightarrow h(x) = Ax + B \equiv \text{ باشد } \text{affine} \text{ باید } h(x)$$

شرایط لازم و کافی بهینه (KKT) برای بهینه‌سازی محدب:

$$L(x, \lambda, \mu) = f(x) + \lambda^T h(x) + \mu^T g(x) = f(x) + \sum_{i=1}^{m} \lambda_i h_i(x) + \sum_{j=1}^{p} \mu_j g_j(x) \rightarrow \text{ لاگرانژین مسئله } \text{تابع}$$

بردار (μ^*, λ^*, x^*) پاسخ بهینه ضرایب لاگرانژ بهینه برای مسئله بهینه‌سازی محدب می‌باشند، اگر و تنها اگر شرایط

Subject :

Year . Month . Date .

شرایط زیر برقرار باشد :

1) $h(x^*)=0\ ,\ g(x^*)\le 0$ → Primal Feasibility

2) $\mu^* \geq 0$ → dual Feasibility

3) $\nabla_x L(x,\lambda,\mu) = \nabla f(x^*) + \lambda^T \nabla h(x^*) + \mu^T \nabla g(x^*) = 0$ → Lagrangian optimality

4) $\mu^{*T} g(x^*) = 0$ → Complementary slackness

Duality :

$$\min\ f_0(x) \quad \text{s.t.}\quad f_i(x)\le 0 \quad i=1,\dots,m \qquad h_i(x)=0 \quad i=1,\dots,p$$

→ $D = \bigcap_{i=1}^{m} \operatorname{dom} f_i \bigcap_{i=1}^{p} \operatorname{dom} h_i$

→ $x \in D \subseteq R^m$

→ optimal value $= p^*$

مسئله دوگان : dual مسئله اولیه (primal) ← مسئله بهینه‌سازی بدون قید + مسئله بهینه‌سازی convex

Lagrangian:

تعریف می‌کنیم :

$$L(x,\lambda,\mu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \mu_i h_i(x)$$

Lagrangian dual function:

$$g(\lambda,\mu) = \inf_{x\in D} L(x,\lambda,\mu)$$

نسبت به x

inf: یعنی بزرگترین کران پایین تابع؛ اگر تابع مینیمم نداشته باشد آن را به عنوان جواب قبول می‌کنیم! → $\inf e^{-x} = 0$

به دنبال مقدار تابع به ازای x هستیم

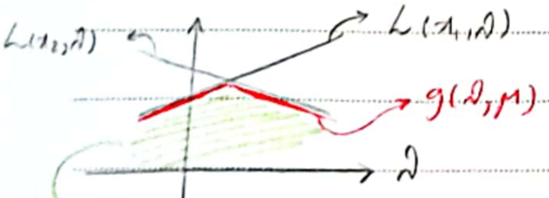
$$= \inf_{x\in D} \left(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \mu_i h_i(x)\right)$$

نکته : تابع $g(\lambda,\mu)$ یک تابع concave یا مقعر می‌باشد. یک تابع خطی از λ_i و μ_i می‌باشد (یک تابع affine می‌باشد)

دو تابع affine هر کدام مقعر هستند؛ inf این تابع مقعر خواهد بود

← اثبات مقعر بودن $g(\lambda,\mu)$:

برای اثبات مقعر بودن یک حالت خاص را در نظر می‌گیریم و با ترسیم آن مقعر بودن را ثابت می‌کنیم



x_1 و x_2 feasible باشند و μ را نداریم! نمودار به شکل زیر خواهد بود :

$hyp(f) = \{(x,r)\in D\times R \mid f(x)\ge r\}$ → f is concave iff hyp(f) convex

↳ hypograph → فضای زیر f

مجموعه‌ای از نقاط

→ $hyp(\inf_{x_1,x_2} L(x,\lambda)) = hyp(L(x_1,\lambda)) \cap hyp(L(x_2,\lambda))$

Subject :

Year . Month . Date .

همانطور که گفته شد تمام جملات داخل int نسبت به λ و μ affine هستند، پس hyp آنها محدب می‌باشد؛

اشتراک چند دامنه محدب، محدب می‌باشد، نتیجه تست‌ها با دامنه محدب بودن یک شرط خوب بود.

مرتبط با قبلی: $epi f = \{(x,r) \in D \times R \mid f(x) \le r\} \rightarrow f$ convex iff epi f ~~concave~~ convex

اگر جواب‌ها دقیق برقرار شود، پاسخ مسئله dual با پاسخ یکی باشد

حال می‌خواهیم یک کران از درگیری‌های $g(\lambda,\mu)$ را بدست بیاوریم، اولیه بیان خواهیم کرد.

Lower bound property: → همان پاسخ بهینه مسئله بهینه‌سازی اولیه

$g(\lambda,\mu) \le p^*$ if $\lambda \ge 0$

اثبات: فرض کنید $\tilde{x}$ یک پاسخ شدنی و $\lambda \ge 0$ باشد:

$$f_0(\tilde{x}) \ge L(\tilde{x},\lambda,\mu) \overset{\text{اثبات}}{\Rightarrow} f_0(\tilde{x}) + \sum_{i=1}^{m} \underbrace{\lambda_i}_{\ge 0} \underbrace{f_i(\tilde{x})}_{\le 0} + \sum_{i=1}^{p} \mu_i \underbrace{h_i(\tilde{x})}_{0} \le f_0(\tilde{x})$$

we know: $L(\tilde{x},\lambda,\mu) \ge \inf_{x \in D} L(x,\lambda,\mu) = g(\lambda,\mu)$

از آنجایی که رابطه بالا به ازای هر $\tilde{x}$ شدنی برقرار است، بنابراین برای $\tilde{x}$ شدنی دقیق‌تر $f_0(x)$ نیز برقرار است

در نتیجه: $f_0(x^*) = p^* \ge g(\lambda,\mu)$

The dual problem (Lagrange dual problem): $\max\ g(\mu,\lambda)$ s.t. $\lambda \ge 0$

مسئله‌ی دوگان convex است زیرا $g(\lambda,\mu)$ مقعر بود و $\lambda \ge 0$ نیز محدب می‌باشد، پس کل مسئله محدب می‌باشد

حل آن بهترین حد پایین برای p^* را بدست می‌دهد (d^* = dual مقدار بهینه مسئله)

Duality gap: $p^* - d^*$ → $d^* \le p^*$ weak duality

→ $d^* = p^*$ strong duality → برای مسئله‌ی convex تحت شرایط اسلاتر برقرار است

و همچنین برای برخی مسائل non-convex نیز می‌تواند برقرار باشد.

به این شرایط اسلاتر معادل منفی بودن، بهینه یافته شده است.

Batus

Subject :

Year . Month . Date .

Slater's Constraint qualification (CQ) مسئله اولیه : $\min f_0(x)$

$$\text{s.t } f_i(x) \le 0 \quad i=1,\ldots,m$$
$$Ax = b$$

if $\exists \tilde{x} \in \text{int } D : f_i(\tilde{x}) < 0$, $i=1,\ldots,m$, $Ax=b$ → در این صورت مسئله convex اصطلاحا strictly feasible است و شرایط CQ برقرار است.

بنابراین برای مسئله convex با احراز شرایط CQ، strong duality برقرار خواهد بود، $p^* = d^*$

حال می‌خواهیم این موضوع را بررسی کنیم که آیا مسئله‌ی dual و مسئله‌ی اصلی (با فرض اینکه $d^* = p^*$ باشد) جواب (λ^*, μ^*) ، $\{x, \lambda, \mu\}$

هر مسئله یا صرفا تقاطع ... چنانکه گفته شد شرایط لازم و کافی برای بهینه بودن مسئله، شرایط KKT می‌باشد!

پس اگر جواب بهینه را در مسئله dual در شرایط KKT صدق کند نشان می‌دهیم $\{\mu, \lambda, x\}$ جواب بهینه ...

متغیر خاص ... تبدیل بهینه مسئله اصلی را پیدا می‌کنیم! حال این موضوع را بررسی می‌کنیم:

1) در حالت ابتدا در مسئله dual گفته شد که λ^*, μ^* جواب بهینه است، از شرایط KKT استفاده می‌کنیم.

2) $g(\lambda, \mu) = \inf_{x \in D} \{L(x, \lambda, \mu)\} \rightarrow x^* \in \underset{x \in D}{\text{Argmin}}\, L(x, \lambda^*, \mu^*)$ (I)

بنابراین اگر $\tilde{x}$ جواب به دست آمده از رابطه فوق شرط feasibility و CS (شرایط KKT) را برآورده کند، در این صورت جواب بهینه مسئله primal

خواهد بود. یک حالت خاص این است که مسئله primal جواب بهینه یکتا داشته باشد (مثلا strictly convex باشد)

در این صورت هر مسئله dual با جواب بهینه مسئله primal را به دست خواهد داد: در غیر این صورت حتما باید شرایط KKT

اطلاعات کند. در شرایط مسئله dual را به صورت $\{\mu, \lambda, x\}$ ... اگر جواب مسئله $\tilde{x}$ است به (λ^*, μ^*) ، $L(x, \lambda^*, \mu^*)$ را مینیمم کند $\tilde{x}$ نباشد!

(به صورت یکتا)
(منحصر به فرد)

Balus

Subject : Strong duality

Year . Month . Date .

$$p^* = f_0(x^*) = g(\lambda^*,\mu^*) = \inf_x \left\{ f_0(x) + \sum_{i=1}^{m} \lambda_i^* f_i(x) + \sum_{i=1}^{p} \mu_i^* h_i(x) \right\}$$

$$= f_0(\tilde{x}) + \sum_{i=1}^{m} \underbrace{\lambda_i^*}_{\ge 0} \underbrace{f_i(\tilde{x})}_{\le 0} + \sum_{i=1}^{p} \mu_i^* \underbrace{h_i(\tilde{x})}_{0} \le f_0(\tilde{x}) = f_0(\tilde{x})$$

اگر پاسخ بهینه یکتا باشد

(*) اگر پاسخ بهینه داشته باشیم: 1) feasible آنگاه $f_0(\tilde{x}) = f_0(x^*)$

2) در شرایط CS صدق کند ← $\sum_{i=1}^{m} \lambda_i f_i(\tilde{x}) = 0$

در این شرایط λ^*, μ^*

(*) بنابراین $\tilde{x}$ نیز یک پاسخ بهینه برای مسئله primal خواهد بود! (λ^* و μ^* مسئله dual نیز باید شرایط برابری فراهم کنند)

Ex. $\min \sum_{i=1}^{n} f_i(x_i)$

s.t $a^T x = b \longrightarrow a = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix}, x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$

$f_i(x_i) = \frac{1}{2} c_i x_i^2 \rightarrow$ These are strictly convex and $a^T x = b$ is convex $\Rightarrow \sum f_i(x_i)$ is strictly convex

مسئله را با روش dual حل می‌کنیم:

$$g(\mu) = \min_x \left\{ \sum_{i=1}^{n} \frac{1}{2} c_i x_i^2 + \mu(b - a^T x) \right\} = \mu b + \min_x \left\{ \sum_{i=1}^{n} \frac{1}{2} c_i x_i^2 - \mu \sum a_i x_i \right\}$$

$$= \mu b + \min_x \left\{ \sum_{i=1}^{n} \left\{ \frac{1}{2} c_i x_i^2 - \mu a_i x_i \right\} \right\} = \mu b + \sum_{i=1}^{n} \min \left\{ \frac{1}{2} c_i x_i^2 - \mu a_i x_i \right\}$$

با مشتق‌گیری: $x_i^* = \frac{a_i \mu}{c_i}$

$$g(\mu) = \mu b + \sum_{i=1}^{n} \left[\frac{1}{2} \frac{a_i^2}{c_i} \mu^2 - \frac{a_i^2}{c_i} \mu^2 \right] = \mu b - \frac{\mu^2}{2} \underbrace{\sum_{i=1}^{n} \frac{a_i^2}{c_i}}_{t} = b\mu - \frac{1}{2} t \mu^2$$

$$\max g(\mu) = \max\; b\mu - \frac{1}{2} t \mu^2 \Rightarrow \mu^* = \frac{b}{t} \rightarrow x_i^* = \frac{a_i \mu^*}{c_i} = \frac{a_i b}{c_i t}$$

ارتباط نقطه زینی تابع لاگرانژ و مقادیر بهینه مسئله primal و dual:

نقطه زینی: جفت بردار $x^* \in X$ و $z^* \in Z$ یک نقطه زینی برای تابع φ به شمار می‌رود:

$$\varphi(x^*, z) \le \varphi(x^*, z^*) \le \varphi(x, z^*) \quad \forall x \in X, \; \forall z \in Z$$ → (نسبت به x مینیمم و نسبت به z ماکزیمم است)

Subject :

Year . Month . Date .

به طور معادل می‌توان گفت $z^* \in Z$ و $x^* \in X$ یک نقطه زینی برای تابع φ است اگر و فقط اگر

$$\sup_{z\in Z} \varphi(x^*,z) = \varphi(x^*,z^*) = \inf_{x\in X} \varphi(x,z^*)$$

لم: اگر جفت (x^*,z^*) یک نقطه زینی برای تابع φ باشد، آنگاه تساوی minimax برقرار است:

$$\sup_{z\in Z}\inf_{x\in X} \varphi(x,z) = \inf_{x\in X}\sup_{z\in Z} \varphi(x,z)$$

اثبات: ① Minimax inequality (در همه شرایط برقرار است): $\forall \bar z \in Z \quad \varphi(x,\bar z) \le \sup_{z\in Z} \varphi(x,z)$

inf

$$\Rightarrow \inf_{x\in X} \varphi(x,\bar z) \le \inf_{x\in X}\sup_{z\in Z} \varphi(x,z) \quad \text{for all } \forall \bar z\in Z$$

sup

$$① \Rightarrow \sup_{\bar z\in Z}\inf_{x\in X} \varphi(x,\bar z) \le \inf_{x\in X}\sup_{z\in Z} \varphi(x,z)$$

Ex. $f(x,y) = x+y$, دامنه $S = \{(x,y) \mid x=0,1,2,\ y=0,1,2,\ |x-y|<2\}$

① $\inf_x \sup_y (f) = 1+2 = 3$ ② $\sup_y \inf_x (f) = 0+1 = 1 \Rightarrow \sup\inf \le \inf\sup$ ادامه

اگر (x^*,z^*) نقطه زینی باشد، $\inf_x \sup_z \varphi(x,z) \le \sup_z \varphi(x^*,z) = \varphi(x^*,z^*) = \inf_x \varphi(x,z^*) \le \sup_z\inf_x \varphi(x,z)$

② $\Rightarrow \inf_x\sup_z \varphi(x,z) \le \sup_z\inf_x \varphi(x,z)$

① and ② : $\inf\sup \varphi(x,z) = \sup\inf \varphi(x,z)$ if (x^*,z^*) زینی باشد

نتیجه: (x^*,z^*) نقطه زینی تابع φ است اگر و فقط اگر تساوی minimax برقرار باشد، و x^*، z^* به ترتیب پاسخ مسائل زیر باشند:

$$x^* = \operatorname{Argmin}_x \sup_z \varphi(x,z)$$

$$z^* = \operatorname{Argmax}_z \inf_x \varphi(x,z)$$

Subject :

Year . Month . Date .

اثبات: اگر x^* و z^* پاسخ مسائل بهینه سازی فوق باشند داریم:

$$\sup_{z}\inf_{x} \varphi(x,z) = \inf_{x\in X} \varphi(x,z^*) \leq \varphi(x^*,z^*) \leq \sup_{z} \varphi(x^*,z) = \inf_{x}\sup_{z} \varphi(x,z)$$

عکس قضیه: اگر $(\bar{x},\bar{z})$ نقطه زینی تابع φ باشد، در متن فوق ثابت شد تساوی minimax برقرار است، همچنین

بر اساس نامساوی ② تساوی در سمت راست نامساوی برقرار خواهد شد، در نتیجه آن x^* و z^* پاسخ مسائل بهینه سازی

فوق خواهند بود.

معادل تساوی minimax باشد

نتیجه: هنگامی (x^*,λ^*,μ^*) یک نقطه زینی برای تابع لاگرانژین $L(x,\lambda,\mu)$ است اگر و فقط اگر x^* و (λ^*,μ^*)

به ترتیب پاسخ بهینه مسئله primal و dual باشند و همچنین strong duality برقرار باشد.

$$L(x,\lambda,\mu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \mu_i h_i(x) \rightarrow \sup_{\lambda\geq 0,\,\mu} L(x,\lambda,\mu) = \begin{cases} f_0(x) & h_i(x)=0,\ f_i(x)\leq 0 \ \text{feasible} \\ \infty & \end{cases}$$

$$\rightarrow \inf_{x}\sup_{\lambda} L(x,\lambda,\mu) = p^* \Rightarrow x^* = \arg\min \sup L(x,\lambda,\mu) \longrightarrow$$ جواب مسئله primal

$$\sup_{\lambda\geq 0,\,\mu}\inf_{x} L(x,\lambda,\mu) = d^* \Rightarrow (\lambda^*,\mu^*) = \arg\max_{\lambda\geq 0,\,\mu} \inf L(x,\lambda,\mu) \rightarrow$$ جواب مسئله dual

اگر strong duality برقرار باشد، یعنی $d^* = p^*$ پس: $\inf_{x}\sup_{\lambda\geq 0,\,\mu} L(x,\lambda,\mu) = \sup_{\lambda\geq 0,\,\mu}\inf_{x} L(x,\lambda,\mu)$

بنابراین (x^*,λ^*,μ^*) یک نقطه زینی برای تابع لاگرانژین خواهد بود.

Batus

Subject :

Year . Month . Date .

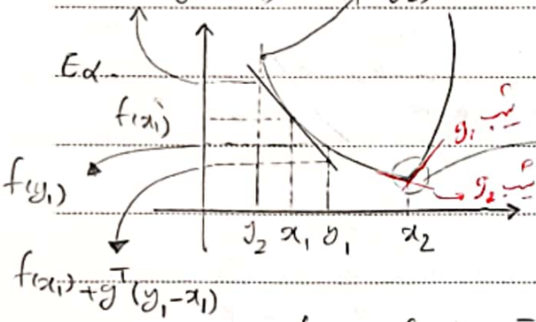
Subgradian (زیرگرادیان):

نوعی تعمیم به روش گرادیان برای تابع مشتق‌ناپذیر

تعریف: بردار زیرگرادیان (مشتق‌ناپذیر) برای تابع $f: X \subseteq \mathbb{R}^n \to \mathbb{R}$ در x است اگر: $f(y) \geq f(x) + g^T(y-x)$

$f(y_2) \geq f(x_1) + g^T(y_2 - x_1)$

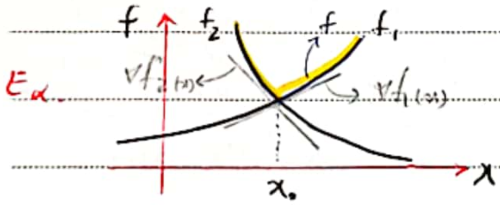
← برای توابع محدب و مشتق‌پذیر، زیرگرادیان همان گرادیان می‌شود!



شیب‌های g_1 و g_2 زیرگرادیان می‌باشند! ← در این نقطه، تابع مشتق‌ناپذیر است

زیرگرادیان همان گرادیان می‌باشد ← $f(y_1) \geq f(x_1) + g^T(y_1 - x_1)$

Ex.



$f = \max(f_1, f_2)$

$g =$ زیرگرادیان f

$$\begin{cases} f_1(x) > f_2(x) \to g = \nabla f_1(x) \\ f_2(x) > f_1(x) \to g = \nabla f_2(x) \\ f_1(x) = f_2(x) \to g \in [\nabla f_2(x_0), \nabla f_1(x_0)] \end{cases}$$

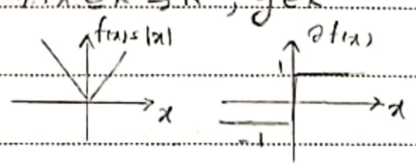
← اگر یک تابع بیش از یک Subgradient داشته باشد، مجموعه‌ای به اسم Subdifferential تعریف می‌شود.

Subdifferential (زیردیفرانسیل):

به مجموعه زیرگرادیان‌های تابع f در x، زیردیفرانسیل f در x گفته می‌شود و با $\partial f(x)$ نشان می‌دهند.

$\partial f(x) = \{g : f(z) \geq f(x) + g^T(z-x)\}$
$\forall z \in X$

Ex. $f: X \subseteq R^n \to R$, $g \in R^n$



الگوریتم زیرگرادیان برای بهینه‌سازی مسائل محدب:

$f(x): R^n \to R$ (محدب) → می‌خواهیم Min این تابع را بدون محدودیت پیدا کنیم → initial x_0 → $x_{k+1} = x_k - \alpha_k g_k$

Step size

یک زیرگرادیان تابع f در مرحله k در x_k می‌باشد؛ از آنجا که ممکن است در هر مرحله بهبود نداشته باشیم، این روش مانند روش گرادیان نزولی وجود ندارد.

Batus

Subject :

Year . Month . Date .

تعریف: بهترین جواب در k مرحله اول: $f_k^{best} = \min_{i=1,2,\dots,k} f(x_i)$

حال می‌خواهیم بررسی کنیم که step size را چگونه انتخاب کنیم:

1) $\alpha_k = \alpha$ ثابت

2) طول گام ثابت: $\|x_{k+1} - x_k\| = \gamma \ , \ \alpha_k = \frac{\gamma}{\|g_k\|_2}$

3) مجموع مربعات محدود و مجموع نامحدود: $\sum_{k=1}^{\infty} \alpha_k^2 < \infty \ , \ \sum_{k=1}^{\infty} \alpha_k = \infty \quad \lim_{k\to\infty} \alpha_k = 0$

Ex. $\alpha_k = \frac{1}{k}$

4) مجموع و مجموع مربعات نامحدود: $\sum_{k=1}^{\infty} \alpha_k^2 = \infty \ , \ \sum_{k=1}^{\infty} \alpha_k = \infty$ Ex. $\alpha_k = \frac{1}{k^{0.5}}$

سوال: آیا تضمین همگرایی برای توابع محدب در الگوریتم زیرگرادیان وجود دارد؟

سوال: آیا همگرایی به مقدار بهینه اتفاق می‌افتد؟

برای پاسخ دادن به این ۲ سوال، یک سری مفروضات را در نظر می‌گیریم:

1) پاسخ بهینه x^* وجود دارد به طوری که: $f(x^*) = f^* = \inf_x f(x) > -\infty$

2) نرم زیرگرادیان تابع و در همه نقاط محدود باشد: $\|g\|_2 \le G \quad \forall g \in \partial f \quad \forall x \in R^n$

3) فاصله نقطه اولیه (شرایط اولیه) از نقطه بهینه (x^*) محدود باشد: $\|x_0 - x^*\| \le R$

اگر نرم شرایط اولیه محدود انتخاب شود، نرم x^* باید محدود باشد.

تحلیل همگرایی الگوریتم زیرگرادیان:

$$\|x_{k+1} - x^*\|_2^2 = \|x_k - \alpha_k g_k - x^*\|_2^2 = (x_k - \alpha_k g_k - x^*)^T (x_k - \alpha_k g_k - x^*)$$

$$= \|x_k - x^*\|^2 - 2\alpha_k g_k^T (x_k - x^*) + \alpha_k^2 \|g_k\|^2$$

(ادامه در صفحه بعد)

atus

Subject :

Year . Month . Date .

حال از تعریف subgradient استفاده می‌کنیم:

$$f(y) \geq f(x) + g^T(y-x) \Rightarrow g_k^T(x^*-x_k) \leq f(x^*) - f(x_k)$$

$$\Rightarrow \|x_k - x^*\|^2 + 2\alpha_k g_k^T(x^*-x_k) + \alpha_k^2\|g_k\|^2 \leq \|x_k - x^*\|^2 - 2\alpha_k\left(f(x_k) - f(x^*)\right) + \alpha_k^2\|g_k\|^2$$

روی همه مراحل از 1 تا k سیگما اعمال می‌کنیم:

$$\Rightarrow \sum_{i=1}^{k} \|x_{i+1} - x^*\|^2 \leq \sum_{i=1}^{k} \|x_i - x^*\|^2 - 2\sum \alpha_i\left(f(x_i) - f(x^*)\right) + \sum_{i=1}^{k} \alpha_i^2 \|g_i\|^2$$

ساده کردن

$$\rightarrow \|x_{k+1} - x^*\|^2 \leq \|x_1 - x^*\|^2 - 2\sum_{i=1}^{k}\alpha_i\left(f(x_i) - f(x^*)\right) + \sum_{i=1}^{k}\alpha_i^2\|g_i\|^2$$

(1) $\|x_{k+1} - x^*\|^2 \geq 0 \Rightarrow \|x_1 - x^*\|^2 - 2\sum_{i=1}^{k}\alpha_i\left(f(x_i) - f(x^*)\right) + \sum_{i=1}^{k}\alpha_i^2\|g_i\|^2 \geq 0$

(2) $\sum_{i=1}^{k} \alpha_i\left(f(x_i) - f(x^*)\right) \geq \sum_{i=1}^{k} \alpha_i\left(f_k^{best} - f^*\right)$

در رابطه بالا این مقدار جایگذاری شد

$$\Rightarrow \underbrace{\|x_1 - x^*\|^2}_{\leq R^2} - 2\sum_{i=1}^{k}\alpha_i\left(f_k^{best} - f^*\right) + \sum_{i=1}^{k}\alpha_i^2\|g_i\|^2 \geq 0$$

$$\Rightarrow \left(2\sum_{i=1}^{k}\alpha_i\right)\left(f_k^{best} - f^*\right) \leq R^2 + \sum \alpha_i^2\|g_i\|^2 \Rightarrow f_k^{best} - f^* \leq \frac{R^2 + \sum_{i=1}^{k}\alpha_i^2\|g_i\|^2}{2\sum_{i=1}^{k}\alpha_i}$$

در حالت کلی و به ازای step size های مختلف چه اتفاقی می‌افتد؟

(1) α ثابت: $f_k^{best} - f^* \leq \frac{R^2 + G^2\sum_{i=1}^{k}\alpha^2}{2\sum_{i=1}^{k}\alpha} = \frac{R^2 + kG^2\alpha^2}{2k\alpha} \quad (\|g_i\|^2 \leq G)$

if $k \to \infty$: $f_k^{best} - f^* \leq \frac{G^2\alpha}{2}$

بنابراین در این حالت الگوریتم همگرا می‌شود اما f_k^{best} لزوماً به f^* نمی‌رسد! بلکه در یک باندی از f^* قرار می‌گیرد!

(2) طول گام ثابت: $\alpha_k = \frac{\gamma}{\|g_k\|}$: $f_k^{best} - f^* \leq \frac{R^2 + \sum_{i=1}^{k}\frac{\gamma^2}{\|g_i\|^2}\|g_i\|^2}{2\sum_{i=1}^{k}\frac{\gamma}{\|g_i\|}} \leq \frac{R^2 + \sum_{i=1}^{k}\gamma^2}{\frac{2}{G}\sum_{i=1}^{k}\gamma} = \frac{R^2 + k\gamma^2}{\frac{2k\gamma}{G}}$

if $k \to \infty$: $f_k^{best} - f^* \leq \frac{\gamma G}{2}$

Batus

Subject :

Year . Month . Date .

(3) $\sum_{k=1}^{\infty} \alpha_k = \infty \,,\; \sum_{k=1}^{\infty} \alpha_k^2 < \infty \rightarrow f_k^{best} - f^* \leq \dfrac{R^2 + G^2 \sum_{i=1}^{k} \alpha_i}{2 \sum \alpha_i} \,,\; \text{if } k \to \infty \;\; f_k^{best} - f^* \leq 0$

$\Rightarrow f_k^{best} \leq f^*$

f^* is The minimum

$\Rightarrow f_k^{best} = f^*$

(4) $\sum_{k=1}^{\infty} \alpha_k = \infty \,,\; \sum_{k=1}^{\infty} \alpha_k^2 = \infty \,,\; \lim_{k \to \infty} \alpha_k = 0 \xrightarrow[\text{در نهایت}]{\text{می‌توان اثبات کرد}} f_k^{best} = f^*$

ویژگی های subgradient : ① روش مرتبه اول

② نسبت به روش هایی مانند نیوتن (مرتبه دوم) کندتر است.

مزایا :

(۱) قابل استفاده برای توابع مشتق ناپذیر

(۲) محاسبات ساده

(۳) قابلیت مناسب برای استفاده در الگوریتم های توزیع شده (غیر متمرکز)

(٤) قابل استفاده در حالت مقید

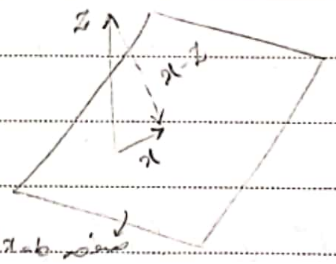
projected subgradient Method : (روش گرادیان تصویر شده)

$\text{Min } f(x)$

$\text{s.t. } x \in D \,,\; D \in R^n \,,\; \text{convex} \;\; \rightarrow x_{k+1} = Pr_D(x_k - \alpha_k g_k)$

نقطه ای از مجموعه D که نزدیک ترین فاصله را از بردار $x_k - \alpha_k g_k$ دارد انتخاب می کنیم.

$\Pi_D(x) = pr_D(x) = \underset{\tilde{x} \in D}{\text{Arg min}} \, \| x - \tilde{x} \|$



Ex. $\text{Min } f(x)$ $\quad x_{k+1} = \underbrace{Pr_{Ax=b}(\underbrace{x_k - \alpha_k g_k}_{z})}_{x}$

$\text{s.t. } Ax = b$

$\text{Min } \|x - z\|^2 \rightarrow$ هدف بدست آوردن عبارت x جایی است که فاصله x از بردار z کمینه شود جایی که

$\text{s.t. } Ax = b$ x متعلق به صفحه $Ax = b$ باشد

مسئله با ضرایب لاگرانژ حل می‌کنیم :

$L(x, \lambda) = \|x - z\|^2 + \lambda^T (Ax - b)$

$\dfrac{\partial L}{\partial x} = 2(x - z) + A^T \lambda = 0 \xrightarrow{\times A} 2(Ax - Az) + AA^T \lambda = 0$ ④

$= 2(b - Az) + AA^T \lambda = 0 \Rightarrow \lambda = 2(AA^T)^{-1}(Az - b)$

Batus

باید معکوس پذیر باشد ⇐ قیود مستقل باشند (یکدیگر را نقض نکنند).

Subject :

Year . Month . Date .

$$x = z - A^T(AA^T)^{-1}(Az-b)$$ با جایگذاری h در رابطه ⓐ خواهیم داشت:

$$x \leftrightarrow x_{k+1} \ , \ z \leftrightarrow x_k - \alpha_k g_k$$

$$x_{k+1} = x_k - \alpha_k g_k - A^T(AA^T)^{-1}(A(x_k - \alpha_k g_k) - b) = x_k - \alpha_k g_k - A^T(AA^T)^{-1}(\cancel{b} - A\alpha_k g_k - \cancel{b})$$

$$= x_k - \alpha_k g_k - A^T(AA^T)^{-1}(-A\alpha_k g_k) = x_k - \alpha_k [I - A^T(AA^T)^{-1}A] g_k$$

$$\Rightarrow x_{k+1} = x_k - \alpha_k \underbrace{[I - A^T(AA^T)^{-1}A] g_k}_{\to \Pr_{Ax=0}(g_k)}$$

در واقع تصویر g_k بر فضای صفر $Ax=b$ می‌باشد! (★) $N(A)$

این رو باید از x_k حذف کنیم!

(★) we know: $x = z - A^T(AA^T)^{-1}(Az-b) \xrightarrow{set} \begin{cases} z \to g_k \\ Ax=b \to 0 \end{cases} \to x = [I - A^T(AA^T)^{-1}A] g_k$

پس (★) تصویر g_k بر روی $Ax=0$ می‌باشد!

$$\Rightarrow x_{k+1} = x_k - \alpha_k \Pr_{Ax=0}(g_k)$$

حال می‌خواهیم همگرایی این روش را بررسی کنیم.

همگرایی: با فرضیات step size مشابه زیرگرادیان:

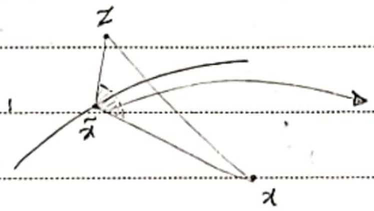
$$\begin{cases} z_{k+1} = x_k - \alpha_k g_k \\ x_{k+1} = \Pi_D(z_{k+1}) \end{cases}$$

اپراتور تصویرسازی (Π_D) یک اپراتور Nonexpansive است؛ به این معناست که:

$$\|\Pi_D(z) - x\| \le \|z - x\|$$

$x \in D \to$ Convex

اثبات:



به علت convex بودن D، این زاویه همواره از $90°$ بیشتر خواهد بود.

برای اثبات همگرایی:

$$\|x_{k+1} - x^*\|_2^2 = \|\Pi_D(z_{k+1}) - x^*\|^2 \le \|z_{k+1} - x^*\|^2$$

$x^* \in D$

بنابراین از این مرحله به بعد اثبات همگرایی مشابه حالت الگوریتم زیرگرادیان بدون تغییر خواهد بود و در شرایط

مشابه با فرض convex بودن مجموعه قیود الگوریتم زیرگرادیان تصویرشده همگرا خواهد شد.

Subject :

Year .   Month .   Date .

حل مسئله dual به کمک روش گرادیان تصویر شده،

Primal: $\min f_0(x)$

$\boxed{\text{s.t. } f_i(x) \leq 0} \quad i=1,\dots,m$ ②

Dual: $\max g(\lambda) \equiv \min -g(\lambda)$

s.t. $\lambda \geq 0$ $\quad \boxed{\text{s.t. } \lambda \geq 0}$ ① $\quad , \quad g(\lambda) = \inf\limits_{x} L(x,\lambda)$

تصویرسازی راحت‌تر می‌باشد ⇐ ① نسبت به ② ساده‌تر می‌باشد

منفی

داشتیم، $L(x,\lambda) = f_0 + \sum_{i=1}^{m} \lambda_i f_i(x)$

قبلاً نشان دادیم $g(\lambda)$ نسبت به λ، concave است، بنابراین $-g(\lambda)$ تابعی convex است.

حال باید subgradient عه $g(\lambda)$ را حساب کنیم،

$h_k \in \partial(-g(\lambda_k)) \Rightarrow \lambda_{k+1} = [\lambda_k - \alpha_k h_k]_+$

این یعنی مقادیر مثبت را نگه داریم و مقادیر منفی را صفر می‌کنیم

$(x)_+ = \max(x,0)$

$\ast \; -g(\lambda) = -\inf\limits_{x} L(x,\lambda) = \sup\limits_{x} -L(x,\lambda) \Rightarrow -g(\lambda) = \sup\limits_{x} -(f_0(x) + \lambda_1 f_1(x) + \dots + \lambda_m f_m(x))$

زیرگرادیان $-L(x,\lambda)$: $h(-L(x,\lambda)) = -[\underbrace{f_1(x)}_{\frac{\partial g}{\partial \lambda_1}}, \dots, \underbrace{f_m(x)}_{\frac{\partial g}{\partial \lambda_m}}] \quad \forall x$

برای اعمال sup را لحاظ کنیم $x = x^*$ باید قرار داده شود

$\Rightarrow h_k = -[f_1(x^*(\lambda_k)), \dots, f_m(x^*(\lambda_k))]^T$

Decentralized optimization:

فرض می‌کنیم مجموعه‌ای از عامل‌ها یا Node ها (بعضی به عنوان عامل یا Node) دارای تابع هدف f_i ، $i=1,\dots,n$ و متغیر تصمیم

گیری x_i ، $i=1,\dots,n$ می‌باشند. مانع از آن می‌شود که f_i

هر تابع به x_i ربط داشته باشد ← Complicating variable (متغیر تداخلی)

هدف حل مسئله‌های بهینه‌سازی با شکل زیر می‌باشد (به عنوان نمونه):

① $\min\limits_{x_1,\dots,x_n,y} \sum_{i=1}^{n} f_i(x_i, y)$

این ۲ باعث می‌شوند که f_i ها به هم مرتبط باشند

② $\min\limits_{x_1,\dots,x_n} \sum_{i=1}^{n} f_i(x_i) \quad \text{s.t.} \quad \sum_{i=1}^{n} x_i \leq c$

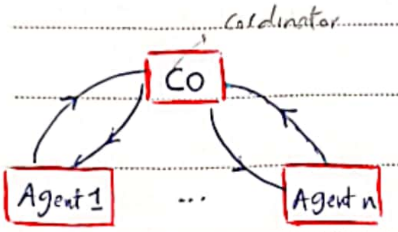
Batus

Subject :

Year . Month . Date .

دلایل عدم حل متمرکز مسائل بزرگ:
۱- گستردگی جغرافیایی و عدم امکان تبادل اطلاعات در حجم گسترده
۲- حجم محاسبات مسئله متمرکز و مسئله مقیاس‌پذیری
۳- اطلاعات خصوصی کاربران

ساختاری که برای بهینه‌سازی غیرمتمرکز ارائه می‌دهیم به شکل زیر می‌باشد:

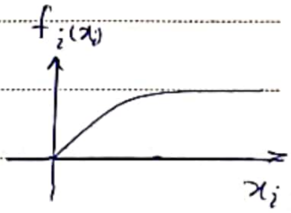


مثال: برای مسئله ② Resource Allocation:

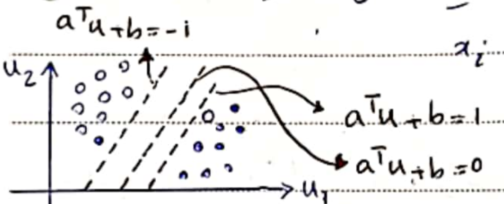
$x_i \to$ مقدار تخصیص (کالا، ارزان، ...) به عامل i

$\sum_{i=1}^{n} x_i \le C \to$ محدودیت منابع

f_i Utility function $\to$ میزان ترجیح (رضایت) عامل i از دریافت x_i $\to$ معمولاً این تابع، صعودی و مقعر می‌باشد:



مثال: برای مسئله ①: حل غیرمتمرکز مسئله SVM:



هدف این است که ما می‌خواهیم margin را تا حد ممکن بزرگ بدست آوریم به طوری که تعداد دیتاهایی که به طور غلط دسته‌بندی شده‌اند حداقل باشد. داده‌ها را به صورت زیر تعریف می‌کنیم:

data points : $\{(z_j, y_j), j = 1, \dots, p\}$

بردار feature داده j ام $\to$ label داده : $\circ = y_j = 1$

$z_j \in \mathbb{R}^2$ در این مثال : $\bullet = y_j = -1$

مسئله بهینه‌سازی به صورت کلی به شکل زیر می‌باشد:

$$\min_{a,b} \frac{\lambda}{2}\|a\|^2 + \sum_{j=1}^{p} \max(0, 1 - y_j(a^T z_j + b))$$

index set $\to \bigcup_{i=1}^{m} S_i = \{1, \dots, p\}$

حال فرض می‌کنیم که مجموعه داده‌ها در m محل مختلف توزیع شده‌اند: $\{(z_l, y_l), l \in S_i\} \to$ مجموعه داده‌های i ام

در این صورت مسئله بهینه‌سازی به شکل زیر است:

$$\min_{a,b} \sum_{i=1}^{m} \frac{\lambda}{2m}\|a\|^2 + \sum_{l \in S_i} \underbrace{\boxed{\max(0, 1 - y_l(a^T z_l + b))}}_{f_i(x),\ x = (a,b)}$$

$$= \min_{x_i} \sum_{i=1} f_i(x)$$

Decomposition in Optimization:

مسائل جداپذیر:

$$\underset{x_1,x_2}{\text{Min}}\ f_1(x_1)+f_2(x_2) \equiv \underset{x_1\in C_1}{\text{Min}}\ f_1(x_1) + \underset{x_2\in C_2}{\text{Min}}\ f_2(x_2)$$
$\text{s.t. } x_1\in C_1, x_2\in C_2$

به جز اپراتور جمع (+) این مسئله برای سایر اپراتورهای غیر نزولی نیز جداپذیر است.

اپراتورهای Min و Max نیز غیر نزولی اند یعنی اگر اعداد بزرگتر شوند Min و Max آنها کاهش نخواهد یافت!

$$\underset{\substack{x_1\in C_1\\x_2\in C_2}}{\text{Min}}\ \psi(f_1(x_1),f_2(x_2)) \equiv \psi\left(\underset{x_1\in C_1}{\text{Min}}\ f_1(x_1), \underset{x_2\in C_2}{\text{Min}}\ f_2(x_2)\right)$$

مثلاً این تقسیم برای اپراتور تفریق (−) برقرار نیست:

Ex. $\underset{\substack{0\le x_1\le 1\\1\le x_2\le 2}}{\text{Min}}\ (x_1^2-x_2^2)=-4 \rightsquigarrow \left(\underset{0\le x_1\le 1}{\text{Min}}\ x_1^2\right)-\left(\underset{1\le x_2\le 2}{\text{Min}}\ x_2^2\right)=-1\neq -4$

مسائل با متغیر تکمیلی (Complicating/Coupling Variable): مسئله‌ی جداپذیر می‌باشد

$$\underset{x=(x_1,x_2,y)}{\text{Min}}\ f(x)=f_1(x_1,y)+f_2(x_2,y)=\underset{y}{\text{Min}}\ \boxed{\underset{x_1,x_2}{\text{Min}}\ f(x_1,x_2,y)} \rightarrow$$

Primal Decomposition:

ایده این است که ابتدا بهینه‌سازی به متغیرهای تخصیص در سطح پایین (عامل / Nodeها) انجام می‌شود و سپس حاصل به سطح بالاتر (هماهنگ کننده) ارسال می‌شود تا بهینه‌سازی نسبت به متغیر تکمیلی انجام شده و در نهایت f بهینه می‌گردد.

P_1: $\underset{x_1}{\text{Min}}\ f_1(x_1,y)=\phi_1(y)$ P_2: $\underset{x_2}{\text{Min}}\ f_2(x_2,y)=\phi_2(y)$

حل مسئله در سطح بالاتر به شکل زیر حل می‌شود: $\underset{y}{\text{Min}}\ \phi_1(y)+\phi_2(y)$

به این نوع جداسازی primal decomposition می‌گویند چرا که Co (Coordinator) یک متغیر primal را تعیین می‌کند.

* برای اینکه مسئله ارتباط بین عامل‌ها در سطح پایین و Co تسهیل شود می‌توانیم الگوریتمی پیشنهاد دهیم که صرفاً مقادیر عددی بین عامل‌ها و Co منتقل شود و همچنین Privacy توابع هدف عامل‌ها حفظ گردد.

Subject :

Year . Month . Date .

فرض کنید زیرمسائل P_1 و P_2 توسط عامل‌ها حل شده است.

الگوریتم primal decomposition : ۱) شروع با مقدار اولیه y_0 و $k=0$

۲) $g_1^k \in \partial\phi_1(y_k)$, $g_2^k \in \partial\phi_2(y_k)$

۳) به روزرسانی متغیر پیچیده کننده توسط CO به روش زیرگرادیان: y_k به روش‌های استاندارد بیان شده انتخاب می‌شود.

$y_{k+1} = y_k - \alpha_k (g_1^k + g_2^k)$

۴) $k = k+1$

۵) بازگشت به مرحله ۲ (تا شرط توقف)

Dual Decomposition:

ایده اصلی معادل‌سازی مسئله با تعریف متغیرهای جدید و اضافه کردن قید (اجماع)

$$\text{Min } f(x) = f_1(x_1,y) + f_2(x_2,y) \equiv \text{Min } f_1(x_1,y_1) + f_2(x_2,y_2) \quad \text{s.t. } y_1 = y_2$$

لاگرانژ مسئله معادل: $L(x_1,y_1,x_2,y_2,\vartheta) = f_1(x_1,y_1) + f_2(x_2,y_2) + \vartheta^T(y_1-y_2)$

$$= \underbrace{\left[f_1(x_1,y_1) + \vartheta^T y_1\right]}_{L_1(x_1,y_1,\vartheta)} + \underbrace{\left[f_2(x_2,y_2) - \vartheta^T y_2\right]}_{L_2(x_2,y_2,\vartheta)}$$

بنابراین برای بدست آوردن تابع dual کافی است زیرمسائل زیر را حل کنیم:

Dual Subproblems: $\begin{cases} g_1(\vartheta) = \inf\limits_{x_1,y_1} L_1(x_1,y_1,\vartheta) \\ g_2(\vartheta) = \inf\limits_{x_2,y_2} L_2(x_2,y_2,\vartheta) \end{cases}$ $\Rightarrow$ Dual function: $g(\vartheta) = g_1(\vartheta) + g_2(\vartheta)$

Dual Problem: $\text{Max } g(\vartheta) = \text{Max } g_1(\vartheta) + g_2(\vartheta)$

ϑ توسط CO در سطح بالاتر

حال برای حل الگوریتمی این مسئله (جهت حفظ حریم خصوصی عامل‌ها و تسهیل ارتباطات) هر عامل لازم است یک

زیرگرادیان از تابع $g_2(\vartheta)$ را محاسبه کند و به CO ارسال کند و همچنین CO لازم است در هر مرحله آخرین مقدار

Batus

Subject :

Year . Month . Date .

به‌روزرسانی شده λ را محاسبه و برای عامل‌ها ارسال کند

حال به بررسی زیرگرادیان $-g_2(\lambda)$ می‌پردازیم:

$i=1$: $-g_1(\tilde{\lambda}) = -\inf_{x_1,y_1} \left(f_1(x_1,y_1)+\tilde{\lambda}^T y_1\right)$

$= -f_1(x_1^*(\tilde{\lambda}), y_1^*(\tilde{\lambda})) - \tilde{\lambda}^T y_1^*(\tilde{\lambda}) \geq -f_1(x_1^*(\lambda), y_1^*(\lambda)) - \tilde{\lambda}^T y_1^*(\lambda) + \lambda^T y_1^*(\lambda) - \lambda^T y_1^*(\lambda)$

$= \underbrace{-f_1(x_1^*(\lambda), y_1^*(\lambda)) - \lambda^T y_1^*(\lambda)}_{-g_1(\lambda)} + (-y_1^*(\lambda))(\tilde{\lambda}-\lambda)$

$\Rightarrow -g_1(\tilde{\lambda}) \geq -g_1(\lambda) + (-y_1^*(\lambda))(\tilde{\lambda}-\lambda)$ → طبق تعریف subgradient:

$-y_1^*(\lambda) \in \partial(-g_1(\lambda))$

به طور مشابه داریم: $y_2^*(\lambda) \in \partial(-g_2(\lambda))$

الگوریتم dual decomposition:

۱) مقداردهی اولیه λ_0 و $k=0$

۲) حل زیرمسئله‌های عامل‌ها:

$(x_1^*(\lambda_k), y_1^*(\lambda_k)) = \underset{x_1,y_1}{\mathrm{Argmin}} \left(f_1(x_1,y_1) + \lambda_k^T y_1\right)$

$(x_2^*(\lambda_k), y_2^*(\lambda_k)) = \underset{x_2,y_2}{\mathrm{Argmin}} \left(f_2(x_2,y_2) - \lambda_k^T y_2\right)$

ارسال $y_1^*(\lambda_k)$ و $y_2^*(\lambda_k)$ به CO:

۳) به‌روزرسانی متغیر dual توسط CO:

$\lambda_{k+1} = \left[\lambda_k - \alpha\left(y_2^*(\lambda_k) - y_1^*(\lambda_k)\right)\right]_+$

۴) $k=k+1$

۵) برو به مرحله ۲ تا شرط توقف

حال به بررسی یک مثال می‌پردازیم.

Subject :

Year . Month . Date .

نکات: • در هر مرحله یک حد پایین از جواب بهینه را خواهیم داشت: $g(\lambda) \le f^*$

• در مراحل الگوریتم پاسخ‌ها نزد ما feasible نیستند $\equiv y_1 \neq y_2$ ؛ اگر بخواهیم در مراحل میانی مقادیر شدنی

از متغیرهای y_1 و y_2 داشته باشیم؛ برای مثال، تصویرسازی بر مجموعه $y_1 = y_2$: $\hat{y} = \dfrac{y_1+y_2}{2}$

در این صورت داریم: $f^* \le f_1(x_1,\hat{y}) + f_2(x_2,\hat{y})$

که باعث بهبود پاسخ و یا عدم تغییر آن خواهد شد

می‌توان مقادیر x_1، x_2 را هم توسط بهینه‌سازی بر حسب $\hat{y}$ به روز کرد: $\hat{x}_i = \underset{x_i}{\text{Argmin}}\ f_i(x_i,\hat{y}) \quad i=1,2$

ادامه

$$f^* \le f_1(\hat{x}_1,\hat{y}) + f_2(\hat{x}_2,\hat{y}) \le f_1(x_1,\hat{y}) + f_2(x_2,\hat{y})$$

همگرایی الگوریتم dual decomposition و بازیابی متغیرهای بهینه primal:

از آنجا که مسئله dual نسبت به λ محدب می‌باشد، اگر α_k در روش زیرگرادیان به شکل مناسب انتخاب شود در

این صورت خواهیم داشت: $k \to \infty \Rightarrow \lambda_k \to \lambda^*$ مسئله dual را بهینه‌سازی می‌کند!

حال اگر مسئله بهینه‌سازی محدب باشد و شرایط CQ برقرار باشد، در این صورت duality gap برابر صفر

خواهد بود و λ^* همان ضریب لاگرانژ بهینه مسئله خواهد بود.

متغیرهای primal چگونه به دست می‌آیند؟ ← اینطوری!

$$\begin{cases} \{x_1^*(\lambda_k), y_1^*(\lambda_k)\} = \underset{x_1,y_1}{\text{Argmin}}\ f_1(x_1,y_1) + \lambda_k^T y_1 \\ \{x_2^*(\lambda_k), y_2^*(\lambda_k)\} = \underset{x_2,y_2}{\text{Argmin}}\ f_2(x_2,y_2) + \lambda_k^T y_2 \end{cases}$$

از آنجا که λ_k به سمت λ^* همگرا می‌شود، مقادیر $\{x_1^*, y_1^*\}$ و $\{x_2^*, y_2^*\}$ نیز همگرا خواهند شد.

اگر مسئله بهینه‌سازی صرفاً convex باشد، در این صورت از آنجا به متغیرهای x_i^* و y_i^*، $i=1,2$ نیز دست‌یافتنی نیستند.

Subject :

Year . Month . Date .

بنابراین الگوریتم لزوماً به مقدار بهینه دسترسی x_i^* و y_2^* همگرا نخواهد شد و به تبع آن شرط $y_1=y_2$ نیز در نهایت لزوماً برقرار نخواهد بود.

اما اگر مسئله بهینه‌سازی strictly convex باشد، آنگاه پاسخ x_i^* و y_2^* یکتا می‌باشد و در این صورت الگوریتم
$i=1,2$
به این مقدار همگرا شده و شرط $y_1=y_2$ نیز در نهایت برقرار خواهد شد.

مقایسه روش‌های PD و DD :

	Primal	Dual
	محاسبه تابع $\phi_i(y)$ و دسترسی محاسبه $g_i^k \in \partial\phi_i(y)$	محاسبه $y_2(\lambda_2)$ از حل بهینه‌سازی با امکان حل عددی
	نیاز به محدب بودن ϕ_i ها برای وجود subgrad و همگرایی	مسئله dual یک مسئله محدب است و D همگرا می‌شود
	تعیین یک متغیر اصلی بهینه‌سازی تکراری توسط CO به طور مستقیم	→ Virtual price تنظیم متغیر بهینه‌سازی تکراری به طور غیرمستقیم و از طریق قیمت مجازی
	مقادیر میانی در همه مراحل الگوریتم	مقادیر میانی تنها در صورت همگرایی و محدب اکید بودن مسئله یا تمهیدات بیان شده (تقریب‌سازی)

Decomposition with Constraints

Minimize $f_1(x_1)+f_2(x_2)$

s.t. $x_1 \in C_1 , x_2 \in C_2$

$h_1(x_1)+h_2(x_2) \le 0 \quad (h_i : R^n \to R^p)$ تعداد قیدها

در ادامه با روش Decomposition ، Primal به طور مستقیم این مسئله را حل می‌کنیم.

Subject :

Year . Month . Date .

Primal decomposition:

برای $t \in R^p$ را در نظر می‌گیریم:

$$\text{Minimize } f_1(x_1) + f_2(x_2)$$

$$\text{s.t. } x_1 \in C_1 ,\ x_2 \in C_2$$

$$h_1(x_1) \le t ,\ h_2(x_2) \le -t$$

حال تنها مشکل این است که مقدار t به صورت بهینه تعیین شود که این کار توسط CO انجام خواهد شد:

حل زیر مسئله توسط عامل‌ها

$$\begin{cases} P_1 : \underset{x_1 \in C_1,\ h_1(x_1) \le t}{\text{minimize}}\ f_1(x_1) \ \rightarrow\ \phi_1(t) \\ P_2 : \underset{x_2 \in C_2,\ h_2(x_2) \le -t}{\text{minimize}}\ f_2(x_2) \ \rightarrow\ \phi_2(t) \end{cases}$$

حل مسئله توسط CO:

$$\underset{t}{\text{minimize}}\ (\phi_1(t) + \phi_2(t))$$

ϕ_1 و ϕ_2 باید محدب باشند

حال مانند حالات قبل به یک الگوریتم عددی مبتنی بر گرادیان نیاز داریم. نیازی به ارسال توابع ϕ_1 و ϕ_2 از طرف عامل‌ها به CO نخواهد بود و فقط مقادیر عددی سابگرادیان های ϕ_1 و ϕ_2 در هر مرحله برای CO ارسال می‌شود و CO براساس روش سابگرادیان مقدار t را به روز کرده و برای عامل‌ها ارسال می‌کند.

الگوریتم Primal Decomposition: ← حل زیر مسائل P_1 و P_2 توسط عامل‌ها و بدست آوردن توابع $\phi_1(t), \phi_2(t)$

1) مقداردهی اولیه t_0 و $k=0$

2) محاسبه
$$\begin{cases} d_k^1 \in \partial\phi_1(t_k) \\ d_k^2 \in \partial\phi_2(t_k) \end{cases}$$

3) به روز رسانی t_k: $t_{k+1} = t_k - \alpha_k(d_k^1 + d_k^2)$

4) $k = k+1$

5) برو به مرحله 2 تا شرط توقف

یعنی h_i و f_i و C_i برای $i=1,2$ محدب باشند!

برای مسائل محدب خواهیم داشت:

$$\phi_i(t) = \sup_{\lambda_i \ge 0}\ \inf_{x_i \in C_i} \left(f_i(x_i) + \lambda_i^T (h_i(x_i) \mp t)\right) \rightarrow \begin{cases} d_k^1 = -\lambda_1^*(t_k) \\ d_k^2 = \lambda_2^*(t_k) \end{cases}$$

($-$ for $i=1$, $+$ for $i=2$)

حال می‌خواهیم ببینیم که آیا می‌توانیم از رابطه‌ی بالا سابگرادیان $\phi_i(t)$ را بدست آوریم؟!

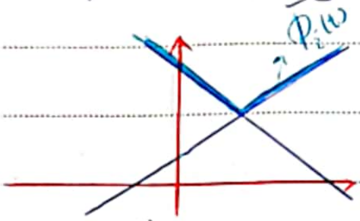
Batus

Subject :

Year . Month . Date .

در ابتدا دو نقطه را با این می‌سنجیم،

۱) تابع $\phi_2(t)$ محدب می‌باشد زیرا sup روی یک خط از t می‌باشد به شرط اینکه شیب بودن آن ثابت باشد؛



۲) همانطور که در رابطه $\phi_i(t)$ دیده می‌شود تنها ضریب t، λ_i می‌باشد، بنابراین،

$d_k^1 = -\lambda_1^*(t_k)$

$d_k^2 = \lambda_2^*(t_k)$

proof: $\phi_1(\tilde{t}) = \sup_{\lambda_1 \geq 0} \inf_{x_1 \in C_1} \left(f_1(x_1) + \lambda_1 (h_1(x_1) - t)\right) \geq \inf_{x_1 \in C_1} \left(f_1(x_1) + \lambda_1^{*T}(t) \times (h_1(x_1) - \tilde{t})\right)$

$= \inf_{x_1 \in C_1} \left(f_1(x_1) + \lambda_1^{*T}(t) \times (h_1(x_1) - \tilde{t}) + \lambda_1^{*T}(t)t - \lambda_1^{*T}(t)t\right)$

$= \inf_{x_1 \in C_1} \left(f_1(x_1) + \lambda_1^{*T}(t)(h_1(x_1) - t) + \lambda_1^{*T}(t)t - \lambda_1^{*T}(t)\tilde{t}\right) = \phi_1(t) + (-\lambda_1^{*T}(t))(\tilde{t} - t)$

$\phi_1(\tilde{t}) \geq \phi_1(t) + (-\lambda_1^{*T}(t))(\tilde{t} - t) \Rightarrow \quad \in \partial\phi_1(t)$

بطور مشابه: $\lambda_2^*(t) \in \partial\phi_2(t) \rightarrow d_k^1 = -\lambda_1^*(t_k)\,,\; d_k^2 = \lambda_2^*(t_k)$

Dual Decomposition:

Minimize $f_1(x_1) + f_2(x_2)$ $\rightarrow$ $L(x_1, x_2, \lambda) = f_1(x_1) + f_2(x_2) + \lambda^T(h_1(x_1) + h_2(x_2))$

s.t $x_1 \in C_1, x_2 \in C_2$ $\qquad = (f_1(x_1) + \lambda^T h_1(x_1)) + (f_2(x_2) + \lambda^T h_2(x_2))$

$h_1(x_1) + h_2(x_2) \leq 0$

حل زیرمسائل توسط عامل‌ها

$$\begin{cases} P_1: \text{minimize } f_1(x_1) + \lambda^T h_1(x_1) \rightarrow g_1(\lambda) \\ \quad \text{s.t } x_1 \in C_1 \\ P_2: \text{minimize } f_2(x_2) + \lambda^T h_2(x_2) \rightarrow g_2(\lambda) \\ \quad \text{s.t } x_2 \in C_2 \end{cases} \Rightarrow g(\lambda) = g_1(\lambda) + g_2(\lambda)$$

co تنها: Maximize$_{\lambda \geq 0}$ $g(\lambda)$

Subject :
Year . Month . Date .

الگوریتم dual decomposition: ۱) مقداردهی اولیه λ_0 و $k=0$

۲) حل زیرمسائل توسط عامل‌ها و به دست آوردن: $d_k^1 \in \partial g_1(\lambda_k)$, $d_k^2 \in \partial g_2(\lambda_k)$

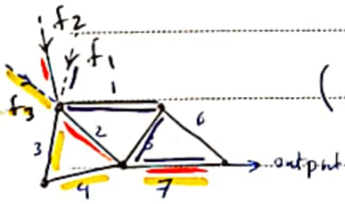
۳) $\lambda_{k+1} = [\lambda_k - \alpha_k(d_k^1 + d_k^2)]_+$ ۴) $k=k+1$ ۵) برو به مرحله ۲ تا شرط توقف

حال می‌خواهیم ببینیم d_k^1 و d_k^2 به چه شکل خواهند بود!؟ $d_k^i = -h_i(x_i^*(\lambda_k))$ $i=1,2$

Meaning: $-h_1(x_1^*(\lambda_k)) - h_2(x_2^*(\lambda_k)) \in \partial(-g(\lambda_k))$

← نقطه ضعف روش primal نسبت به dual این است که در روش primal یک متغیر مابین ایشان مشترک است ولی در dual هر agent با یک شبکه متمرکز حل نشد!

مثال ۱) یک شبکه با n (تعداد) agent داریم، می‌خواهیم جریان f_j را داخل شبکه وارد می‌کند. این شبکه دارای m لینک می‌باشد و تابع معرفی (utility) آن به شکل $U_j(f_j)$ تعریف می‌شود که رضایتمندی ≡ بهره agent را از جریانی به وارد شبکه کرده است. (این تابع اکیداً مقعر و صعودی می‌باشد، برای مثال)



ترافیک هر لینک با t_i نمایش داده می‌شود!

t_i = traffic on link i → $\begin{pmatrix} t_1 \\ \vdots \\ t_m \end{pmatrix} = R_{m \times n} \times \begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix}$

مسیر بنفش، مسیر نارنجی، مسیر زرد

$[R]_{ij} = \begin{cases} 1 & \text{flow } j \text{ passes over link} \\ 0 & \text{otherwise} \end{cases}$ → برای گراف بالا و مسیرهای مشخص شده: $R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}_{7 \times 3} \Rightarrow t_7 = f_1 + f_2 + f_3$

منطقی است که باید برای ترافیک لینک محدودیتی باشد، $t \le C$ → ظرفیت لینک‌ها!

مسئله بهینه‌سازی به شکل زیر است:

$$\text{Maximize} \sum_{j=1}^{n} U_j(f_j)$$

$$\text{s.t.} \quad R_{m \times n} f \le C$$

Subject :

Year . Month . Date .

Dual Decomposition: $\underset{\text{s.t } Rf \le c}{\operatorname{Max}} \sum_{j=1}^{n} U_j(f_j) \equiv \underset{\text{s.t } Rf \le c}{\operatorname{Min}} -\sum U_j(f_j)$

$$L(f,\lambda) = \sum_{j=1}^{n} -U_j(f_j) + \lambda^T(Rf - c) = \sum_{j=1}^{n} -U_j(f_j) + \lambda^T\Big(\sum_{j=1}^{n} r_j f_j - c\Big)$$

$$= -\lambda^T c + \sum_{j=1}^{n}\big(-U_j(f_j) + \underbrace{\lambda^T r_j f_j}\big) \rightarrow [\lambda_1 \dots \lambda_m]\begin{bmatrix} r_{1j} \\ \vdots \\ r_{mj} \end{bmatrix} f_j$$

تعبیری از λ_i: هزینه‌ای باید به ازای استفاده از لینک i پرداخته شد.

Subproblem j: $g_j(\lambda) = \inf_{f_j} (\lambda^T r_j f_j - U_j(f_j))$

$$\Rightarrow g(\lambda) = -\lambda^T c + \sum_{j=1}^{n} \inf(\lambda^T r_j f_j - U_j(f_j))$$

مسئله بهینه‌سازی دوگان $\Rightarrow \underset{\lambda \ge 0}{\operatorname{Max}}\ g(\lambda) \rightarrow d(\lambda) \in \partial(-g(\lambda))$ (*)

(*) $d = c - d^1 - d^2 - \dots - d^n$, $d^j(\lambda) = r_j f_j^*(\lambda)$

بدین ترتیب الگوریتم به صورت زیر خواهد بود:

1) λ_0 , $k=0$

2) محاسبه زیرگرادیان‌ها توسط کاربرها: $d_k^j = -r_j f_j^*(\lambda_k) \in \partial(-g_j(\lambda_k))$

3) به روز رسانی λ_k توسط لینک‌ها: $\lambda_{k+1} = [\lambda_k - \alpha_k(c - r_1 f_1^*(\lambda_k) - r_2 f_2^*(\lambda_k) - \dots - r_n f_n^*(\lambda_k))]_+$

$$= [\lambda_k - (c - Rf^*(\lambda_k))]_+$$

4) $k = k+1$

5) برو به مرحله ۲ تا شرط توقف

Batus

Subject :

Year . Month . Date .

Augmented Lagrangian:

تعمیمی بر خود روش lagrange میباشد! فرض کنید مسئله بهینه سازی زیر را داریم:

Minimize $f(x)$

s.t $Ax=b$ $\quad \rightarrow \quad L(x,y)=f(x)+y^T(Ax-b)$

$x\in\mathbb{R}^n$, $A\in\mathbb{R}^{m\times n}$

$f: \mathbb{R}^n \to \mathbb{R}$ and convex

نکته: نیاز به strictly convex بودن تابع f جهت بازیابی متغیرهای primal در الگوریتم dd.

نکته: نیاز به محدود بودن تابع لاگرانژ از پایین بر حسب x برای بدست آوردن تابع dual.

برای اینکه دو شرط بالا حاصل شود، augmented lag $\mathcal{L}$ تعریف می‌شود:

$$L_p(x,y)=f(x)+y^T(Ax-b)+\left(\frac{p}{2}\right)\|Ax-b\|_2^2$$

Penalty parameter $\leftarrow p>0$

تابع لاگرانژ فوق معادل با مسئله بهینه سازی زیر است:

minimize $f(x)+\left(\frac{p}{2}\right)\|Ax-b\|^2$

s.t $Ax=b$

جواب بهینه مسئله فوق با جواب بهینه مسئله اولیه یکسان است!

اضافه کردن $\|Ax-b\|_2^2$ باعث می‌شود که تابع هدف strictly convex شود زیرا خودش اکیداً محدب میباشد و همچنین

برای پیدا کردن inf Lp لازم نیست تابع محدود باشد ... $\|\ \|^2$ ...

اجازه نمی‌دهد که x به سمت $-\infty$ یا $+\infty$ برود! پس در نتیجه ... $f(x)=\begin{cases}0 & x\in C\\ +\infty & x\notin C\end{cases}$

باشد، ... inf تابع $L_p(x,y)$ ... inf تابع $L_p(x,y)$ را ... بدست آورد.

الگوریتم تکراری حل بهینه سازی با یک Augmented Lagrangian:

Method of Multipliers: این الگوریتم ... میباشد!

$$\begin{cases} x^{k+1}=\underset{x}{\text{Argmin}}\ L_p(x,y^k)\\ y^{k+1}=y^k+p(Ax^{k+1}-b)\end{cases}$$

Step size

این شرط ... الگوریتم ... برقرار ... آنها ... میمانند!

شرایط بهینگی:

$$\begin{cases} Ax^*-b=0\\ 0\in\partial f(x^*)+A^Ty^*\end{cases}$$

$0\in \partial L_p(x^{k+1},y^k)=\partial f(x^{k+1})+A^Ty^k+pA^T(Ax^{k+1}-b)$

$=\partial f(x^{k+1})+A^T(\underbrace{y^k+p(Ax^{k+1}-b)}_{y^{k+1}})$

$=\partial f(x^{k+1})+A^Ty^{k+1}$

... چون x^{k+1} ... $L_p(x,y^k)$ ... $\partial L_p(x^{k+1},y^k)$ ...

ضعف بزرگی که این روش دارد این است که نمی‌توانیم مسئله‌ی جمع را Decompose کنیم؛ در واقع چون رابطه $\|Ax-b\|_2^2$

خطی نمی‌باشد نمی‌تواند بین agent ها تقسیم شود!

به زبان ساده، در الگوریتم (MM) مقادیر (x^k, y^k) در هر مرحله کمک الگوریتم شرط بهینگی تابع لاگرانژ را برآورده می‌کنند و

همچنین الگوریتم در نهایت همگرا شده و شرط feasibility $(Ax^*-b=0)$ در نهایت برقرار خواهد شد. اما نقطه ضعف

این روش این است که لاگرانژین Augment شده قابل decompose شدن برای به دست آوردن حل توزیع شده نیست.

برای حل مشکل فوق روشی به عنوان **Alternating Direction Method of Multipliers (ADMM)**:

۲- این روش در فرم استاندارد مسئله خواهیم داشت به صورت زیر حل کند:

$$\min \; f(x) + g(z) \quad \text{s.t.} \quad Ax + Bz = c \;\rightarrow\; x \in R^n,\; z \in R^m,\; A \in R^{p \times n},\; B \in R^{p \times m},\; c \in R^p$$

و f و g محدب بوده

در اینجا نیز مجدداً از Augmented Lagrang استفاده می‌کنیم:

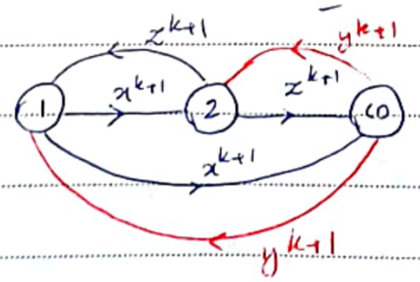
$$L_\rho(x, z, y) = f(x) + g(z) + y^T(Ax + Bz - c) + \left(\frac{\rho}{2}\right)\|Ax + Bz - c\|^2$$

⊛ (در صفحه بعد اثبات شده)

حال الگوریتم ADMM به صورت زیر عمل می‌کند:

$$\begin{cases} x^{k+1} = \underset{x}{\operatorname{Argmin}} \; L_\rho(x, z^k, y^k) \\ z^{k+1} = \underset{z}{\operatorname{Argmin}} \; L_\rho(x^{k+1}, z, y^k) \\ y^{k+1} = y^k + \rho(Ax^{k+1} + Bz^{k+1} - c) \end{cases}$$

بدین ترتیب در الگوریتم فوق x و z به صورت ترتیبی و Alternating به روز می‌شوند. این فرآیند امکان decompose شده را به دست می‌دهد.



Scaled form: ← تغییر متغیر برای ساده‌سازی روش ADMM می‌باشد

$r = Ax + Bz - c \rightarrow$ residual $\qquad u = (\frac{1}{p})y \rightarrow$ scaled dual variable $\Rightarrow y = pu$

$(*) = y^T r + (\frac{p}{2})\|r\|^2 = (\frac{p}{2})\|r + (\frac{1}{p})y\|^2 - (\frac{1}{2p})\|y\|^2$

$= \frac{p}{2}(r + (\frac{1}{p})y)^T(r + (\frac{1}{p})y) = \frac{p}{2}(r^T + \frac{1}{p}y^T)(r + (\frac{1}{p})y) = \frac{p}{2}(\|r\|^2 + \frac{2}{p}y^T r + \frac{1}{p^2}\|y\|^2)$

با جایگذاری رابطه فوق در الگوریتم ADMM خواهیم داشت:

$$x^{k+1} = \underset{x}{\mathrm{Argmin}}\left(f(x) + \frac{p}{2}\|Ax + Bz^k - c + u^k\|^2\right)$$

$$z^{k+1} = \underset{z}{\mathrm{Argmin}}\left(g(z) + \frac{p}{2}\|Ax^{k+1} + Bz - c + u^k\|^2\right)$$

$$u^{k+1} = u^k + Ax^{k+1} + Bz^{k+1} - c \quad \left(y^{k+1} = pu^{k+1} = y^k + p(Ax^{k+1} + Bz^{k+1} - c) = pu^k + p(Ax^{k+1} + Bz^{k+1} - c)\right.$$

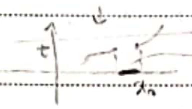
$$\left.\Rightarrow u^{k+1} = u^k + Ax^{k+1} + Bz^{k+1} - c\right)$$

$$r^k = Ax^k + Bz^k - c$$

$$u^k = u^0 + \sum_{j=1}^{k} r^j$$

به زبان ساده، تابعی که سوراخ نداشته باشد، تابعی که همه sublevel set هایش بسته باشند، یعنی $\{x : f(x) \le t\}$ به ازای هر t ای، مجموعه‌ای بسته باشد!

نتایج همگرایی:



این شیب سوراخ ایجاد می‌کند

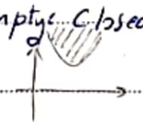
۱) مفروضات: $f: \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ و $g: \mathbb{R}^m \rightarrow \mathbb{R} \cup \{+\infty\}$ توابع f و g بسته (Closed) و محدب سره (proper convex) می‌باشند.

$\exists x \in X \quad f(x) < \infty$

$\forall x \in X \quad f(x) > -\infty$

دقت کنید به این تابع بی‌نهایت می‌گویند شدنی نیست (f خطا باشه شدنی نیست!)

در مجموع این سه شرط معادل این است که epi(f) closed و nonempty و convex باشد.



$epi(f) = \{(x,t) \in \mathbb{R}^n \times \mathbb{R} \mid f(x) \le t\}$

۲) تابع لاگرانژین بسط Augment نشده (L_0) دارای یک نقطه زینی است. معادل وجود جواب بهینه می‌باشد

Subject :

Year . Month . Date .

با فرضیات فوق خواهیم داشت:

(1) $r^k \to 0 \quad k \to \infty$

(2) $y^k \to y^* \quad k \to \infty$

(3) $f(x^k) + g(z^k) \to p^* \quad k \to \infty$

لزوماً x^k و z^k به مقادیر بهینه همگرا نخواهند شد!

proof: $r^k = Ax^k + Bz^k - c \to 0$ if $k \to 0$

x و z ای که در نهایت پیدا می‌شوند هر مقداری در رابطه فوق صدق می‌کنند. اگر A یا B، full rank نباشد ≡ فضای

پوچ آنها تهی نباشد، آنگاه $Ax^k + Bz^k - c$ بی‌نهایت جواب خواهد داشت!

نتیجه ای به همگرایی $f(x^k) + g(z^k) \to p^*$ if $k \to \infty$ بر اساس مورد اتفاق افتاده:

$s^{k+1} = \rho A^T B (z^{k+1} - z^k) \to 0$ if $k \to \infty$

دلیل نتیجه اینجاست که همگرایی s^{k+1} به صفر، همگرایی z^k به z^* را نتیجه نمی‌دهد!

اگر ماتریس $A^T B$ دارای رتبه ستونی کامل (full rank) باشد (A و B هر دو دارای رتبه ستونی کامل باشند)

در این صورت $z^{k+1} \to z^k$ و بنابراین دنباله z^k همگرا خواهد شد و بنابراین بر اساس شرط feasibility ($r^k \to 0$)

و احراز رتبه کامل ستونی A، دنباله x^k نیز همگرا خواهد شد.

همگرایی روش ADMM در عمل: با تعداد تکرار نسبتاً مناسب به جواب قابل قبول می‌رسد ولی برای

همگرایی کامل نیاز به تکرارهای زیادی دارد.

Batus

Subject :
Year . Month . Date .

حال می‌خواهیم روش‌های مطرح شده را بررسی کنیم.

Consider ADMM for generic problem: $\min f(x)$ → bounded below, bounded Subgradient

subject to $x \in C$

ADMM form: take g to be the indicator of C: $\min f(x) + g(z)$ → اگر $z \in C$ باشد، آنگاه مقدار تابع صفر و در غیر این صورت ∞ می‌شود.

subject to $x - z = 0$

Algorithm:

$$\begin{cases} x^{k+1} := \underset{x}{\operatorname{argmin}}\left(f(x) + \left(\frac{\rho}{2}\right)\|x - z^k + u^k\|_2^2\right) \\ z^{k+1} := \Pi_C(x^{k+1} + u^k) \equiv \operatorname{argmin}\left(g(z) + \left(\frac{\rho}{2}\right)\|x^{k+1} - z^{k+1} + u^k\|_2^2\right) \Rightarrow (*) \\ u^{k+1} := u^k + x^{k+1} - z^{k+1} \end{cases}$$

(*) برای آنکه این عبارت Min شود، اگر $z \in C$ باشد و $\|x^{k+1} - z^{k+1} + u^k\|_2^2$ حداکثر شود؟ یعنی $\|\ \|_2^2$ صفر شده باشد پس $u^k + x^{k+1} = z^{k+1}$ باید باشد، ولی ممکن است که $z \in C$ نباشد، پس جواب را بر C تصویر می‌کنیم. اثبات شده است که برای مسائل درجه دو (Quadratic) با تکرار شدن تصویر کردن بر C تصویر کنیم، روندی است که در نهایت به جواب صحیح ختم می‌شود.

مزیت این روش این است که $f(x)$ آزادی بیشتری دارد، یعنی ممکن است اگر $f(x)$ را مستقیماً project کنیم و Min کنیم شاید به جواب نرسد ولی با روش ADMM به جواب برسد.

Lasso (least absolute shrinkage and selection operator)

lasso problem: $\min \frac{1}{2}\|Ax - b\|_2^2 + \lambda\|x\|_1$ → این ترم Regulator است، برای آنکه مسئله ill-posed نشود، باید ... وقتی تعداد observation ها کمتر از تعداد parameter ها باشد، به جواب‌های sparse برسد، وجود $\lambda\|x\|_1$ باعث می‌شود که generalization بیشتری داشته باشد.

در بحث $\|x\|_1$ یا $\|x\|_2$ استفاده می‌کنیم، از بزرگ شدن parameter جلوگیری می‌کنیم، در اینجا از $\|x\|_1$ استفاده می‌کنیم، پارامترهای کم اهمیت را حذف می‌کند؛ $\|x\|_1$ یک چیزی بین این دو می‌باشد، چرا که هم از بزرگ شدن پارامترها جلوگیری می‌کند و بعضی از پارامترها را برابر صفر قرار می‌دهد.

feature selection

Subject :

Year . Month . Date .

$\Rightarrow \|x\|_1 \rightarrow$ Regularization: for ill-posed problems and generalization

$\searrow$ Feature selection

ADMM form: $\quad \text{Min} \ \frac{1}{2}\|Ax-b\|_2^2 + \lambda\|z\|_1$

$\quad \text{s.t} \quad x - z = 0$

ADMM: $L_\rho(x,z,y) = \frac{1}{2}\|Ax-b\|_2^2 + \lambda\|z\|_1 + y^T(x-z) + \rho\|x-z\|_2^2 \rightarrow$ تابع درجه ۲ نسبت به x

$$x^{k+1} := (A^TA + \rho I)^{-1}(A^Tb + \rho z^k - y^k)$$

ترم $\|z\|_1$ باعث میشود آسان این تابع را حل کرد.

$$z^{k+1} := S_{\lambda/\rho}\left(x^{k+1} + \frac{y^k}{\rho}\right)$$ $\rightarrow$ where the soft thresholding operator S is defined as below:

$$y^{k+1} := y^k + \rho(x^{k+1} - z^{k+1})$$

$$S_k(a) = \begin{cases} a-k & a>k \\ 0 & |a| \le k \\ a+k & a<-k \end{cases}$$

Consensus Optimization: want to solve problem with N objective terms: $\text{Min} \sum_{i=1}^{N} f_i(x)$

e.g., f_i is the loss function for ith block of training data

ADMM form: $\quad \text{Min} \sum_{i=1}^{N} f_i(x_i)$

$\quad \text{s.t} \quad x_i - z = 0$

- x_i are local variables
- $x_i - z = 0$ are consistency or consensus constraints
- z is the global variables
- Can add regularization using a $g(z)$ term

ADMM: $L_\rho(x,z,y) = \sum_{i=1}^{N}\left(f_i(x_i) + y_i^T(x_i - z) + \frac{\rho}{2}\|x_i - z\|_2^2\right)$

$$x_i^{k+1} := \underset{x_i}{\text{argmin}}\left(f_i(x_i) + (y_i^k)^T(x_i - z^k) + \frac{\rho}{2}\|x_i - z^k\|_2^2\right)$$

$$z^{k+1} := \frac{1}{N}\sum_{i=1}^{N}\left(x_i^{k+1} + \frac{1}{\rho}y_i^k\right)$$

$$y_i^{k+1} := y_i^k + \rho(x_i^{k+1} - z^{k+1})$$

Subject :

Year . Month . Date .

Federated Learning:

→ Training a machine learning algorithm (e.g: DNN), on multiple local datasets.

→ Without explicitly exchanging data samples (privacy, communicatio cost)

→ Training local models on local datas samples.

→ Sending parameters (e.g the weights and biases of a DNN) from the local nodes to a server

→ Server: aggregates parameters and generate a global model and share it by all nodes.

Core Challenges:

- Expensive Communication
- Systems Heterogeneity (ناهمگونی) → کاربران سیستم‌های مختلفی دارند و هر یک توانایی محاسباتی متفاوتی دارند
- Statistical Heterogeneity → داده‌ها لزوما iid نیستند، یعنی از توزیع‌های مختلفی می‌آیند!
- Privacy Concerns → یک سری تمهیدات را باید در نظر گرفت [illegible]
  و امنیت باید به یک trade off برسیم.

Problem Formulation:

$$\min_{w \in \mathbb{R}^d} f(w) \quad , \quad f(w) \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} f_i(w) \quad , \quad f_i(w) = l(x_i, y_i; w) \quad , \quad F_k(w) = \frac{1}{n_k} \sum_{i \in P_k} f_i(w)$$

n → Number of data points; (x_i, y_i) → Data points; n_k → Number of data points of client k; P_k → Dataset of client k

$$\Rightarrow f(w) = \sum_{k=1}^{K} \frac{n_k}{n} F_k(w)$$

Advantage of Primal Decomposition in Federated Learning: →

→ Non-Convex problem → برخلاف dual decomposition که جواب درستی برای مسئله محدب بود، ولی اینجا لزوما نیاز نیست

→ No personal Variable: Make the update easy at the agent's level →

در روش primal decomposition، $f_i(x_i, y)$ را بر حسب x_i حداقل کردیم تا به $\Phi_i(y) = f_i(x_i^*, y)$ برسیم، سپس $g_i \in \partial\Phi_i(y)$ را

محاسبه کردیم، در Federated learning نقطه شروع [illegible] $F_k(w)$ است!

→ Flexible local updating

→ Low client participation

ادامه در صفحه‌ی بعد آورده شده است

Batus

Subject :

Year . Month . Date .

Algorithm: The K clients are indexed by k; B is the local minibatch size, E is the number of local epoches, and η is the learning rate.

Server execute:

initialize w_0

for each round $t = 1, 2, \dots$ do

$\quad m \leftarrow \max(C \cdot k, 1)$

$\quad S_t \leftarrow$ (random set of m clients)

$\quad$ for each client $k \in S_t$ in parallel do

$\quad\quad w_{t+1}^{k} \leftarrow$ ClientUpdate(k, w_t)

$\quad w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^{k}$

ClientUpdate (k, w): // Run on client k

$\mathcal{B} \leftarrow$ split P_k into batches of size B

for each local epoch i from 1 to E do

$\quad$ for batch $b \in \mathcal{B}$ do

$\quad\quad w \leftarrow w - \eta \nabla \ell(w; b)$

return w to server

$\rightarrow$ (*)

(*) Number of local updates per round: $u_k = E \frac{n_k}{B}$

Decentralized Non-cooperative optimization:

What happens with decomposition solution if the agents are not cooperative?

Let's start with an example: $\max \; (x_1 - \frac{1}{8}x_1^2) + (x_2 - \frac{1}{8}x_2^2)$

$$\text{s.t. } x_1 + x_2 \leq 7$$

The functions are increasing in $[0, 4]$, the constraint is active at the optimal solutions, the Solution obtained by primal dual methods must satisfy the following KKT conditions:

$$L = -x_1 + \frac{1}{8}x_1^2 - x_2 + \frac{1}{8}x_2^2 + \vartheta(x_1 + x_2 - 7)$$

$$\Rightarrow \frac{\partial L}{\partial x_1} = -(1 - \frac{1}{4}x_1) + \vartheta = 0 \;,\; \frac{\partial L}{\partial x_2} = -(1 - \frac{1}{4}x_2) + \vartheta = 0 \;,\; \frac{\partial L}{\partial \vartheta} = x_1 + x_2 - 7 = 0 \Rightarrow x_1 + x_2 = 7$$

The resulting solution: $x_1^* = 3.5$, $x_2^* = 3.5$, $\vartheta^* = \frac{1}{8}$

Let's consider the dual variable ϑ associated with coupling constraint as the "real price" and "ϑx_i" as the corresponding "real payment". Then the payoff of agent i, is given by:

$$U_i(x_i, \vartheta) = x_i - \frac{1}{8}x_i^2 - \vartheta x_i$$

[illegible] agent i، x_i [illegible] [illegible]

Batus

At optimal solution we have : $U_i(x_i^*, \vartheta^*) = x_i^* - \frac{1}{8} x_i^{*2} - \vartheta^* x_i^* = 1.53125$

Consider agent 1 deviates from $x_1^* = 3.5$ to $\hat{x}_1 = 3$ while the strategy of agent 2 is fixed to $x_2^* = 3.5$. Since the constraint is not active any more , we have $\vartheta^* = 0$.

Let's recompute the payoff value of agent 1:

$$U_1(\hat{x}_1, \vartheta^*) = U_1(3,0) = \hat{x}_1 - \frac{1}{8}\hat{x}_1^2 - \vartheta^* \hat{x}_1 = 1.875 > U_1(x_1^*, \vartheta^*) = 1.53125$$

The same situation always happens if the value functions are increasing and the constraint is active and then, one of the agents slighly decrease its value to make its payment equal to zero and earns more profit.

Result : A selfish agent may have incentive to deviate from optimal solution of global optimization.

→ راه حل Mechanism Design می‌باشد ، الگوریتم‌ها تشویق به طراحی مناسب payment function می‌پردازند تا
agentهای خودخواه بهترین x_i را انتخاب کنند که optimal نیز هستند.

حل با این روش Mechanism Design نداریم و با یک non-cooperative game سروکار داریم که فقط به حل آن می‌پردازیم.

→ Non-cooperative multi-agent decision making

→ Each selfish agent only aims to maximizing its own payoff function.

→ The global optimality (social optimality) may be lost in this case.

→ John Nash proposed the concept of Nash Equilibrium to analyze non-coopratine games.

1) we start with a rather general formulation of game ، برنامه‌ریزی ترتیب بازی

2) We define the Nash equilibrium.

3) And then, we propose a decentralized solution to obtain the Nash equilibrium

Game components:

- Set of Players (agents) , $N = \{1, 2, \ldots, N\}$ تعداد بازیکن‌ها برابر n است
- Strategy Sets : $x_n \in X_n$, $\forall n \in N$
- Objective functions: $J_n(x_n, x_{-n})$ $\forall n \in N$, $x_{-n} = [x_1^T \; x_2^T \ldots x_{n-1}^T \; x_{n+1}^T \ldots x_N^T]^T$

→ x_n را طوری تعیین کنیم که J_n حداکثر گردد

Subject :

Year . Month . Date .

Best Response Function:

Each agent wants to minimize its own objective function:

$$BR_n(x_{-n}) = \underset{x_n \in X_n}{\operatorname{argmin}} \; J_n(x_n, x_{-n}) \quad \forall n \in \mathcal{N}$$

However, the result depends on the strategy of other players. →

به خاطر همین موضوع ۲ سوال مطرح می‌شود.

1) How can an agent evaluate best response function to make a decision?

2) What would be the outcome of the game if all the agents follow their best response?

(we need to analyze the behavior of players)

Information and Learning Game:

Naturally the agents do not have any information neither on other agent's objective function nor their current strategies (non-cooperative nature).

Each agent needs at least historical information on other agents strategies to make a decision.

- Such information is available when:

  The game is repeated for some iterations, say $t = 1, 2, \ldots$

  The agents can see others' decisions after each round

  Or a coordinator broadcasts some information on the historical data of the game

The agents can make an estimate from other agents' strategies using historical data.

The most commonly used and realistic estimation is the simplest one, since people usually think simply (again, we are only analyzing the player's behavior). It is called Naïve estimation (Naive expectation):

$\hat{x}_{-i}(t) = x_{-i}(t-1)$ → انتظار داریم استراتژی بعدی agentها همانی باشد که الان دارند، تغییر نکند.

Nash Equilibrium: → سوال اینجاست که آیا این نقطه همیشه وجود دارد

What is the expected final outcome of the game? The Nash Equilibrium (NE) point.

What is the property of NE? No player can better off by unilaterally changing its strategy give the other players' strategies are fixed.

$(x_1^*, x_2^*, \ldots, x_N^*)$ is an NE if $\forall n \in \mathcal{N}, \; \forall x_n \in X_n : J_n(x_n^*, x_{-n}^*) \leq J_n(x_n, x_{-n}^*)$

Subject :

Year . Month . Date .

Simple example: prisoner's Dilemma

prisoner 1 \ prisoner 2	Cooperate	Defect
Cooperate	3,3	0,5
Defect	5,0	1,1

۲ زندانی داریم که قرار است از شدن اعتراف بگیریم: اگر هر ۲ همکاری کنند 3,3 سال، اگر هر ۲ اعتراف کنند (1,1) سال، اگر یکی اعتراف نکند و دیگری نه (0,5) سال حبس می‌شوند. اگر هر ۲ نفر این را بدانند به سمت اعتراف می‌روند، چون اگر خودش هم اعتراف نکند

تابع هدف هر agent جدا

حالت خاصی از مسئله‌ها!

دیدیم به NE می‌رسیم!

Aggregative/Mean_Field game: → تا اینجا فرض کردیم عامل‌ها جدا از هم بوده و باعث شدن نتیجه بشه دیگر

- Objective Function : $J_n(x_n, \sigma(x_N))$ $\forall n \in N$

مفهوم social optimality به تعادل معنا دارد

حالا می‌خواهیم به حالت ترکیب شده یک جنبه‌ی سازنده‌ی عرضه‌ی مسئله بپردازیم. تعادل

تاثیر بقیه عامل‌ها به صورت یک ترم وارد تابع هدف می‌شود

- Aggregative term: $\sigma(x_N) = \frac{1}{N}\sum_{n\in N} x_n$ → ترم تجمیعی ... x ها معلوم باشند، ... داریم:

- Best Response: $x_n^{br} = \underset{y\in X_n}{\arg\min}\ J_n\left(y, \frac{1}{N}y + \frac{1}{N}\sum_{\substack{j\in N\\ j\neq n}} x_j\right)$ $\forall n \in N$

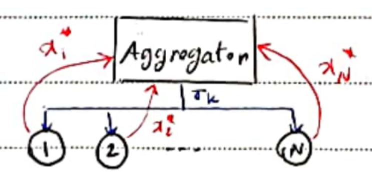
→ Each agent doesn't have direct access to other agent's historical data.

→ There is an aggregator who knows the historical decisions of the agents (more realistic: eg. smart grid, transport or wireless networks, ...)

→ The agents may do not have even information on the structure of aggregative term.

→ The aggregator broadcasts some numerical information on the aggregative term to all the agents.

Game procedure:



Agents: $x_n^*(\sigma_k) = \underset{y\in X_n}{\arg\min}\ J_n(y, \sigma_k)$

Aggregator: $\Lambda(\sigma_k) = \sum_{n\in N}\frac{1}{N}x_n^*(\sigma_k)$

$\sigma_{k+1} = (1-\alpha_k)\sigma_k + \alpha_k\Lambda(\sigma_k)$

Algorithm:

initialization: randomly initialize σ_0, $k \leftarrow 0$

iteration: for $n \in N$

$x_n^*(\sigma_k) \leftarrow \underset{y\in X_n}{\arg\min}\ J_n(y, \sigma_k)$

end

$\Lambda(\sigma_k) \leftarrow \sum_{n\in N}\frac{1}{N}x_n^*(\sigma_k)$

$\sigma_{k+1} \leftarrow (1-\alpha_k)\sigma_k + \alpha_k\Lambda(\sigma_k)$

$k \leftarrow k+1$

آنقدر الگوریتم را اجرا می‌کنیم تا همگرا شود!

we don't expect to reach exactly to NE since each agent neglects its effect in the mean-field term.

هر agent، $\bar{\sigma}_k$ را دریافت می‌کند ولی اثر خود را در آن نمی‌بیند و سپس آن را استفاده می‌کند تا $x_n^* = \arg\min_{y \in X_n} J_n(y, \bar{\sigma}_k)$ را پیدا کند
$\bar{\sigma}_k$ هیچ کاره است، پس نباید انتظار داشته باشیم که به NE همگرا شویم.

The ε-Nash Equilibrium:

$$J_n\left(x_n^*, \frac{1}{N}\sum_{j \in N} x_j^*\right) \leq \min_{x_n \in X_n} J_n\left(x_n, \frac{1}{N}x_n + \frac{1}{N}\sum_{j \in N, j \neq n} x_j^*\right) + \epsilon \quad \forall n \in N$$

این یعنی بهترین پاسخ به اندازه ε از NE فاصله دارد، (اگر agent n اثر $\frac{1}{N}x_n^*$ را در $\bar{\sigma}_k$ ندیده است،

این یعنی $\bar{\sigma}_k$ در بهینه‌سازی دخیل می‌شد، هر agent به اندازه ε می‌توانست سود بیشتری کند.)

نکته: هر چه تعداد N بیشتر باشد، آنگاه ترم $\frac{1}{N}x_n$ کوچکتر شده و تاثیر آن کمتر می‌گردد و این یعنی با صرف نظر کردن

این ترم از $\bar{\sigma}_k$، باز هم همچنان به NE همگرا می‌شود در واقع ε کوچک و کوچکتر می‌شود.

Convergence assumptions:

Assumption on Learning rate: α_k non-increasing, non-summable $\sum_{k=1}^{\infty} \alpha_k = \infty$, square-summable $\sum_{k=1}^{\infty} \alpha_k^2 < \infty$

Assumptions on the objective functions: $J_i(y, \sigma)$ is uniformly Lipschitz continuous

$$J_i(y, \sigma) - J_i(\tilde{y}, \tilde{\sigma}) \leq \underset{\text{ضریب لیپشیتز}}{L}\left(\|y - \tilde{y}\| + \|\sigma - \tilde{\sigma}\|\right)$$

The best-response mappings are non-expansive: $x_i^*(\sigma) - x_i^*(\tilde{\sigma}) \leq \|\sigma - \tilde{\sigma}\|$

Convergence results:

$\sigma_{k+1} = (1-\alpha_k)\sigma + \alpha_k \sigma = \sigma$ ؛ اگر $\Lambda(\sigma_k)$ از جایی به بعد تغییر نکند:

→ Algorithm converges to a fixed point of the aggregation mapping Λ: $\Lambda(\sigma) = \frac{1}{N}\sum_{i=1}^{N} x_i^*(\sigma) = \sigma$

→ The collective set of best-response strategies is an ϵ_N-Nash equilibrium

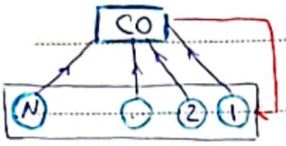
→ ϵ_N is inversely proportional to N!

Subject :

Year . Month . Date .

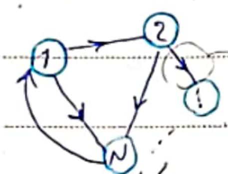
Distributed Optimizations

در ابتدا می‌خواهیم فرق این مبحث با Decentrilized optimization را بیان کنیم ؛ مسائلی که در بخش Decentrilized داشتیم اینطور



بودند که عامل‌ها با coordinator در ارتباط بودند ! و خود عامل‌ها با یکدیگر در ارتباط نبودند !

در Distributed optimization می‌خواهیم نیاز به Coordinator را به طور کامل حذف کنیم و عامل‌ها با ارتباط محلی با یکدیگر



بتوانند مسئله را حل کنند. همچنان قرار نیست که تابع هدف یا اطلاعات شخصی‌شان را ارسال کنند

شبکه توزیعی !

یک بحث پیش‌نیاز :

اجماع (Consensus) : چگونه تعدادی عامل می‌توانند به کمک محاسبات توزیع‌شده و ارتباطات بین خود، روی یک مقدار توافق کنند ؟! (در واقع مسئله این است که چون دیگر Coordinator نداریم که مقدار متغیر تصمیم را تعیین کند، از کجا بفهمیم که همه روی یک مقدار رسیدند و آن مقدار بهینه باشد ؟!) مفهوم اجماع به سوال اول، به رسیدن عامل‌ها به یک توافق می‌پردازد)

حال نیاز داریم که مدل ارتباطی عامل‌ها را به صورت یک گراف تعریف کنیم و براساس آن، بررسی اجماع را روی آن گراف تعریف کنیم :

گراف : مدل ارتباطی عامل‌ها

نودها یال‌ها ماتریس مجاورت

$G(V, E, A)$ — به جهت وجود A

→ directed weighted graph

یال‌ها زیرمجموعه‌ای از تمام حالات عامل‌ها می‌باشد.

تعداد عامل‌ها

(node set) مجموعه نودها : $V = \{1, \dots, m\}$ | (edge set) مجموعه یال‌ها : $E \subseteq V \times V$, $(i,j) \in E$, $i, j \in V$ $i \to j$

← فرض می‌شود از هر نود به خودش یالی وجود ندارد : $(i,i) \notin E \ , \ \forall i \in V$

(Adjacency Matrix) ماتریس مجاورت : $A = [a_{ij}] \in R^{m \times m}$, $\forall (i,j) \in V \times V$ $a_{ij} \geq 0$ (a_j^i)

$a_{ij} > 0$ iff $(j, i) \in E$

a_j^i

$j \xrightarrow{a_{ij} > 0} i$

Batus

انتقال اطلاعات ... از j به i [illegible]

Subject :
Year . Month . Date .

مسیر (path): مجموعه مرتبی از نودها است بطوری که هر جفت متوالی از نودها متعلق به مجموعه E باشند.

$i \to j \to k$

اتصال گراف (connectivity, connected graph): در این صورت به ازای هر $i, j \in V$ مسیری از (j به i یا از i به j) وجود دارد.

Strongly Connected graph:

در صورتی که به ازای هر $i, j \in V$ مسیری از j به i وجود دارد. (یعنی همه به هم مسیر دارند)

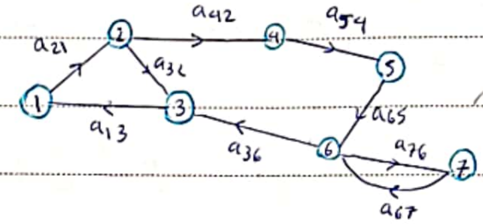
Fully connected graph:

در صورتی که به ازای هر $i, j \in V$ داشته باشیم ($i \neq j$): $(i,j) \in E$

Undirected graph:

گراف G، undirected گفته میشود اگر: $\forall i,j \in E \Rightarrow (j,i) \in E$; $i \rightleftarrows j \equiv i - j$

مثال:



$A = \left[\ \ \right]_{7\times7}$ → Strongly connected directed weighted graph

گراف متغیر با زمان: $G_t = \{V, E(t), A(t)\}$

همسایگی: مجموعه نودهایی که قابلیت ارسال اطلاعات برای نود i در لحظه t را دارند همسایگان نود i در لحظه t میباشند و با مجموعه $N_i(t)$ نمایش داده میشوند:

$N_i(t) = \{ j \in V \mid (j,i) \in E(t) \} \cup \{i\}$ → این یعنی نود i اطلاعات خودش را دارد!

درایه هایش

بردار تصادفی (Stochastic vector): بردار $a \in R^m$ تصادفی است اگر $a_i \geq 0$ داشته باشیم، $\sum_{i=1}^{m} a_i = 1$, $\forall i \in V$

شبیه تابع توزیع احتمال شده میباشد.

ماتریس تصادفی (Stochastic Matrix): ماتریس مربعی $A_{m\times m}$ تصادفی است اگر همه سطرهای آن تصادفی باشند.

* ماتریس مربعی $A_{m\times m}$ تصادفی دوبل (double stochastic) است، اگر A و A^T هر دو تصادفی باشند.

Subject :
Year . Month . Date .

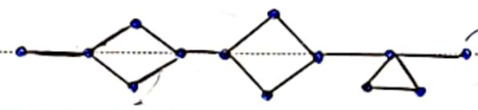
مسئله اجماع (Consensus problem): فرض کنید در گراف $G_t(V,E(t),A(t))$ عامل‌ها (واحدهای محاسباتی) به عنوان Nodeهای گراف و ارتباط بین عامل‌ها از طریق یال‌ها و ماتریس مجاورت مدل شده. حال اگر $x_i(t) \in R^n$ متغیر عامل i در زمان t باشد، آنگاه مسئله اجماع به معنی طراحی (مدل‌سازی) الگوریتم توزیع شده تحت گراف G_t است به طوری که عامل‌ها از طریق تبادل اطلاعات متغیر خود با عامل‌های دیگر و به‌روزرسانی مقدار متغیر خود بر اساس این اطلاعات

$c \in R^n$ و ثابت

در نهایت هر مقدار متغیر به اجماع برسند، به عبارتی: $\lim_{t\to\infty} x_i(t) = c \quad i=1,2,\dots,m$

Peer to peer: نودها کاملاً توزیع شده   شاخص آماری مثل avg

مثال: Computing aggregates in P2P Network



Data Network

هر یک از Nodeها اطلاعاتی را در خود ذخیره کرده است.

← هر Node (برای مثال i) میزان حجم ذخیره داده‌ها را با مقدار مشخص θ_i دارد.

← مقدار θ_i فقط در مکان نود i معلوم است و دسترسی مرکزی به θ_iها وجود ندارد.

← Nodeها از طریق یک شبکه ارتباطی undirected به هم متصل می‌باشند.

صورت مسئله: با یک طراحی یک الگوریتم توزیع شده، مقدار متوسط حجم داده‌های ذخیره شده در کل شبکه را محاسبه کنید. (تخمین بزنید)

$x = \frac{1}{m}\sum_{i=1}^{m}\theta_i \to$ مقدار متوسط داده ذخیره شده در کل شبکه $\equiv \min_{x\in R} \sum_{i=1}^{m}(x-\theta_i)^2$

اگر از این عبارت مشتق بگیریم و برابر صفر قرار دهیم، x برابر $\frac{\sum\theta_i}{m}$ خواهد شد.

$x_i \to x$ تخمین نود i از مقدار x   هدف: $\lim_{t\to\infty} x_i(t) = x$

Subject :

Year .   Month :   Date .

Weighted Averaging Algorithm: (الگوریتم اجماع)

در این الگوریتم هر عامل به روش متوسط وزن دار مقدار متغیر (تخمین) خودش به روز رسانی می‌کند.

$$x_i(k+1) = \sum_{j \in N_i(k)} a_{ij}(k)\, x_j(k)$$ شامل خود i هم می‌باشد

← شروع الگوریتم: $x_i(0)$ را برای $i = 1, \dots, m$ به عنوان مقدار اولیه تعیین می‌کنیم.

← $a_{ij}(k)$ وزنی است که عامل i به اطلاعات دریافتی از عامل j می‌دهد.

← همسایه‌های عامل i و مقادیر وزنی مربوطه ($a_{ij}(k)$) می‌توانند با زمان تغییر کنند.

← وزن‌ها مقادیر غیرمنفی دارند و برای هر عامل i مجموع وزن‌ها برابر ۱ می‌باشد.

$a_{ij} \geq 0$, $a_{ij} > 0$ for $j \in N_i(k)$ and $\sum_{j \in N_i(k)} a_{ij}(k) = 1 \quad \forall i,k$

← ماتریس وزنی، همان ماتریس مجاورت گراف $G_k = (V, E(k))$ با در نظر گرفتن وزن مثبت برای self loop ها است.

$$A(k) = [a_{ij}(k)], \quad a_{ij}(k) = \begin{cases} a_{ij}(k) > 0 & (j,i) \in E(k) \text{ or } j = i \\ 0 & \text{Otherwise} \end{cases}$$

همگرایی الگوریتم اجماع:

در ادامه این شرط را relax تر می‌کنیم ~ periodically strongly connected

مفروضات: ۱) گراف $G_k = (V, E(k))$ strongly connected باشد.

یعنی اگر اجتماع یال‌ها در یک بازه زمانی در نظر بگیریم گراف strongly connected باشد.

یعنی نمی‌توان گفت در یک بازه همه یال‌ها باشند نیاز!

۲) مقادیر مثبت ماتریس وزنی $A(k)$ (به طور یکنواخت) محدود و بزرگتر از صفر می‌باشند

if $a_{ij}(k) > 0 \rightarrow a_{ij} \geq \eta > 0$ از پایین

۳) Node ها از رابطه‌ی زیر مقدار تخمین خود از x را به‌روز رسانی می‌کنند:

$$x(k+1) = A(k)\,x(k) \quad , \quad x(k) = \begin{pmatrix} x_1(k) \\ x_2(k) \\ \vdots \\ x_m(k) \end{pmatrix} \; ; \; \begin{cases} x_i(k) = \sum_{j \in N_i(k)} a_{ij}(k)\, x_j(k) \quad \forall i \\ a_{ij}(k) \geq 0 \;,\; \sum_{j \in N_i(k)} a_{ij}(k) = 1 \end{cases}$$

جمع عناصر سطر i ام ماتریس $A(k)$

$A(k)$ یک ماتریس تصادفی است. در این‌جا فرض شده که $x_i(k)$ یک اسکالر می‌باشد. برای نشان دادن این تعمیم برای دقت که $x_i(k)$ برداری باشد هم برقرار می‌باشد.

قضیه: تحت مفروضات فوق، مقادیر عامل‌ها به یک مقدار اجماعی همگرا می‌شوند به طوری که:

$$\lim_{k \to \infty} x_i(k) = \alpha \quad \forall i \;\rightarrow\; \alpha = \sum_{j=1}^{m} \pi_j\, x_j(0) \quad \begin{cases} \pi_j > 0 \\ \sum_{j=1}^{m} \pi_j = 1 \end{cases}$$

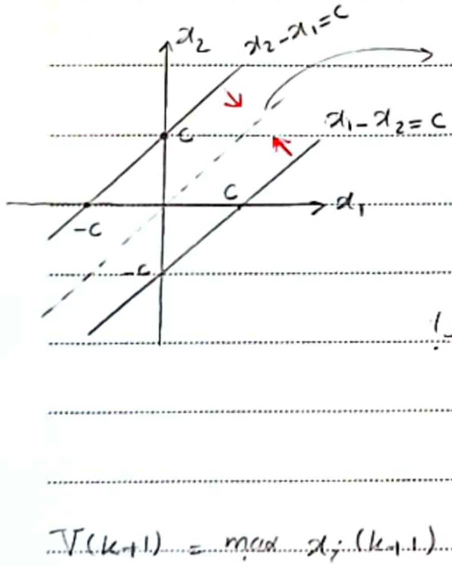
convex combination از شرایط اولیه

اثبات:

سطر i ام از ماتریس $x(k)$ — سطر j ام از ماتریس $x(k)$

تابع لیاپانوف، $V(k) = \max_i x_i(k) - \min_j x_j(k)$

واضح است که تابع $V(k)$ همیشه مثبت است. اگر ثابت کنیم که در طول زمان این تابع نزولی می‌باشد، آنگاه می‌توان اثبات کرد که از یک جایی به بعد x_i ها همه برابر هم خواهند بود. خط مورد علاقه



اگر بخواهیم به صورت شهودی نشان دهیم، فرض کنیم نقطه (x_1, x_2) را داریم.

چیزی که در واقع ما باید اثبات کنیم این است که در طول زمان c به سمت صفر میل کند!

۱) تحت مفروضات بیان شده، $V(k)$ غیرصعودی است:

$$V(k+1) = \max_i x_i(k+1) - \min_j x_j(k+1)$$

$$x(k+1) = A(k)\,x(k)$$

$$i^* = \underset{i}{\mathrm{Argmax}}\; a_i^T(k)\, x(k) \quad , \quad j^* = \underset{j}{\mathrm{Argmin}}\; a_j^T x(k)$$

سطر i ام ماتریس $A(k)$ — $x_i(k+1)$ — سطر j ام ماتریس $A(k)$ — $x_j(k+1)$

Subject :
Year . Month . Date .

② ابتدا تعریف میکنیم $\nabla(k)$ ، از زمان آپدیت شدن x_iها را از زمان s تا k در نظر بگیریم :

$$x(k+1) = A(k)x(k) = A(k)A(k-1)\ldots \underbrace{A(s+1)\overbrace{A(s)x(s)}^{x(s+1)}}_{x(s+2)} = \Phi(k,s)x(s)$$

$$\Phi(k,s) = A(k)A(k-1)\ldots A(s+1)A(s)$$

میخواهیم نشان دهیم اگر $k-s+1$ به اندازه کافی بزرگ باشد، آنگاه همه عناصر ماتریس $\Phi(k,s)$ غیرصفر خواهد بود.

ابتدا به بررسی یک سری نکات می‌پردازیم :

۱) همه عناصر غیرصفر $\Phi(k,s)$ دارای حداقل η^{k-s+1} می‌باشند. (از فرض شبیه به حالت قبل، اگر یک از ماتریس‌های $A(i)$ در یکدیگر ضرب شده باشند و یا با هم جمع شده باشند، آنگاه بقیه درایه‌ها از آن ضرب بزرگتر نیز باشند!)

۲) همه عناصر قطر اصلی $\Phi(k,s)$ مثبت هستند، زیرا ماتریس‌های $A(s)$ ... $A(k)$ همگی دارای عناصر قطری مثبت هستند.

پس اگر سطر اول در ستون اول ضرب شود، درایه اول حداقل غیرصفر است پس قطر اصلی مثبت خواهد داشت!

۳) اگر عنصر غیرصفر (مثبت) (i,j) در یکی از ماتریس‌های $A(t)$ و $s \le t \le k$ وجود داشته باشد، آنگاه عنصر (i,j) در ماتریس $\Phi(k,s)$ نیز غیرصفر است.

حالت اول :
$$\underbrace{\begin{bmatrix} - a_i^T - \end{bmatrix}}_{A:\ \text{غیرصفر}} \underbrace{\begin{bmatrix} | \\ a_j' \\ | \end{bmatrix}}_{B} = \begin{bmatrix} \ \cdot\ \end{bmatrix} \quad (i,j) = a_i^T a_j' \geq AB > 0$$

حالت دوم :
$$\underbrace{\begin{bmatrix} a_i^T \end{bmatrix}}_{B} \underbrace{\begin{bmatrix} | \\ a_j' \\ | \end{bmatrix}}_{A:\ \text{غیرصفر}} = \begin{bmatrix} \ \cdot\ \end{bmatrix} \quad (i,j) = a_i^T a_j' \geq AB > 0$$

Batus

Subject :

Year . Month . Date .

۴) اگر وجود داشته باشد عنصر غیرصفر (i,v) در $A(t)$ ($r \le t \le k$) و همچنین وجود داشته باشد عنصر (v,j) در $A(t)$ ($s \le t < r$)

آنگاه عنصر (i,j) در ماتریس $\Phi(k,s)$ غیرصفر (مثبت) است.

$$\boxed{\cdots\left[\;(i,v)\;\right]\cdots}\;\boxed{\left[\;(v,j)\;\right]} = \left[\;(i,j)\;\right]$$

مثبت مثبت مثبت

$$i\left[\;\cdots\times\cdots\;\right]_{v}\left[\;\times\;\right]^{v} = \left[\;\overset{(i,v)}{\cdot}\;\right]$$

مثبت مثبت

امکان انتقال اطلاعات از Node شماره j به i Node ⇐

با واسطه (از طریق) نود v.

تعداد واسطه‌ها می‌تواند بیشتر از یک باشد: $(i,v)(v,w)(w,j) \equiv (i,j)$

نکته: اگر گراف strongly connected (با n نود) باشد، آنگاه پس از $n-1$ مرحله، همه نودها اطلاعات یکدیگر را خواهند داشت.

نتیجه‌گیری: (برای گراف strongly connected)

$$\left[\Phi(k+(m-1)+1,k)\right]_{ij} \ge \eta^{m-1}$$

لازم به ذکر است که اگر گراف periodically strongly connected باشد، آنگاه اطلاعات نود j با وقفه به نود i ارسال خواهد شد، و
حداکثر بین B مرحله‌ها باید یال داشته باشد

$$E_B = \bigcup_{l=k}^{k+B-1} E_l \equiv (i,j) \in E_k \cup E_{k+1} \cup \dots \cup E_{k+B-1} = E_B$$

یعنی اتصال خودش حداقل یک بار برقرار باشد

اگر بگوییم که در هر $B=10$ تکرار (iteration) یک بار نودهای i و j به هم متصل می‌شوند، پس $(m-1)B$ مرحله

iteration لازم داریم که مطمئن باشیم که اطلاعات نودها به هم ارسال شده باشد.

⇐ B مرحله طول می‌کشد تا اطلاعات هر Node به Node همسایه به طور قطعی منتقل شود، بنابراین $(m-1)B$

Batus

Subject :
Year . Month . Date .

مرحله لازم است تا انتقال اطلاعات از هر Node به Node دیگر میسر شود.

نتیجه برای گراف periodically S.C :

$$\left[\Phi(k+\underbrace{(m-1)B}_{B_0}-1,k)\right]_{ij} \geq \eta^{\overbrace{(m-1)B}^{B_0}}$$

ادامه اثبات همگرایی الگوریتم اجماع (نزولی بودن تابع تپ نسبی $V(k)$) : (حتما به بعد مثال نشان دادیم این تابع می‌تواند

نزولی اکید نباشد و در ادامه می‌خواهیم نشان دهیم که اگر ۲ پریود از سیستم بگذرد حتما نزولی می‌باشد و در نتیجه مقدار تابع

در مرحله دوم از مرحله اول کمک می‌گیریم)

$$x(\overbrace{(k+1)B_0}^{(t+1)}) = \Phi((k+1)B_0-1, kB_0)\, x(\overbrace{kB_0}^{t_0})$$

$S = kB_0$

$k = (k+1)B_0 - 1$

$\Rightarrow k-S+1 = B_0 = (m-1)B$ (به تعداد Node ها) → حداکثر مدت زمانی که طول می‌کشد تا اطلاعات کامل رد و بدل شود

$$x(t+1) = \Phi(t,t_0)\,x(t_0)$$

حال می‌خواهیم ثابت کنیم $\Phi(t,t_0)$ یک ماتریس تصادفی می‌باشد.

نکته : ضرب دو ماتریس تصادفی یک ماتریس تصادفی است :

$A_{m\times m}, B_{m\times m} \rightarrow$ ماتریس‌های تصادفی

$1 = \begin{pmatrix}1\\ \vdots\\ 1\end{pmatrix}$

proof : $\left.\begin{matrix}A1=1\\ B1=1\end{matrix}\right\} \Rightarrow (AB)1 = A(B1) = A1 = 1 \Rightarrow$ AB یک ماتریس تصادفی است

بنابراین ماتریس $\Phi(.,.)$ یک ماتریس تصادفی است.

حال به اثبات نزولی بودن $V(k)$ می‌پردازیم :

$$V(t) = \max_i x_i(t) - \min_j x_j(t)$$

$$V(t+1) = \max_i x_i(t+1) - \min_j x_j(t+1) \rightarrow$$ سپس می‌خواهیم نشان دهیم $V(t+1)$ نسبت به $V(t)$ نزولی است.

حال نشان می‌دهیم $V(t+1)$ نسبت به $V(t_0)$ نزولی است :

Batus

Subject :

Year . Month . Date .

$$i^* = \underset{i}{\mathrm{Argmax}}\ \underbrace{\varphi_i^T x(t_0)}_{x_i(t+1)} \ , \quad j^* = \underset{j}{\mathrm{Argmin}}\ \underbrace{\varphi_j^T x(t_0)}_{x_j(t+1)}$$

$$V(t+1) = \varphi_{i^*}^T x(t_0) - \varphi_{j^*}^T x(t_0) = \varphi_{i^*}^T \Big[\underbrace{1 \max_i x_i(t_0)}_{(1)} + x(t_0) - 1 \max_i x_i(t_0)\Big]$$

$$+ \varphi_{j^*}^T \Big[1 \min_j x_j(t_0) - x(t_0) - \underbrace{1 \min_j x_j(t_0)}_{(1)}\Big]$$

(1) $\underbrace{\varphi_{i^*}^T 1}_{1} \max_i x_i(t_0) - \underbrace{\varphi_{j^*}^T 1}_{1} \min_j x_j(t_0) = \max_i x_i(t_0) - \min_j x_j(t_0) = V(t_0)$

$$\Rightarrow V(t+1) = V(t_0) + \underbrace{\varphi_{i^*}^T \Big[\underbrace{x(t_0) - 1 \max_i x_i(t_0)}_{\le 0}\Big]} + \underbrace{\varphi_{j^*}^T \Big[\underbrace{1 \min_j x_j(t_0) - x(t_0)}_{\le 0}\Big]}$$

این اتفاق به منزله‌ی اجماع رخ داده است

صفر شدن این دو عبارت فقط در حالتی اتفاق می‌افتد که $\boxed{x(t_0) = 1\max_i x_i(t_0) = 1\min_j x_j(t_0)}$ باشد. در غیر این

صورت حداقل یکی از عبارت‌های فوق منفی می‌باشد چرا که عناصر $\varphi_{i^*}^T$ و $\varphi_{j^*}^T$ همگی مثبت هستند.

شرایطی فقط در حالتی اتفاق می‌افتد که اجماع رخ داده باشد، در غیر این صورت → $V(t+1) \le V(t_0)$

تابع $V(t+1)$ نسبت به $V(t_0)$ نزولی است.

نرخ همگرایی:

$$V(t+1) \le V(t_0) + (1\gamma^{B_0})^T \Big[\cancel{x(t_0)} - 1\max_i x_i(t_0) + 1\min_j x_j(t_0) - \cancel{x(t_0)}\Big]$$

$$= V(t_0) - (1\gamma^{B_0})^T \Big[\underbrace{1\max_i x_i(t_0) - 1\min_j x_j(t_0)}_{1V(t_0)}\Big] = V(t_0) - (1\gamma^{B_0})^T 1\, V(t_0) = (1 - m\gamma^{B_0}) V(t_0)$$

$$\Rightarrow 0 \le V(t+1) \le (1 - m\gamma^{B_0}) V(t_0) \quad (*)$$

حال می‌خواهیم رابطه‌ی $V(t_0)$ و $V(t+1)$ را بیابیم: از رابطه‌ی (*) بهره می‌بریم:

Batus

Subject :

Year . Month . Date .

$\gamma = 1 - m\eta^{B_0} \rightarrow \quad V(B_0) \leq \gamma V(0) \qquad k=0$

$V(2B_0) \leq \gamma V(B_0) \leq \gamma^2 V(0) \qquad k=1$

$\vdots$

$V(kB_0) \leq \gamma V((k-1)B_0) \leq \dots \leq \gamma^k V(0) \qquad k \leftarrow k-1$

ضریب صحیح

$\Rightarrow V(kB_0) \leq \boxed{(1-m\eta^{B_0})}^k V(0) \rightarrow V(T) \leq (1-m\eta^{B_0})^{\frac{T}{B_0}} V(0) \rightarrow T = kB_0$

cause $1 - m\eta^{B_0} < 1$

$T \to \infty \Rightarrow V(T) \to 0 \Rightarrow \underset{i}{Max}\, x_i(T) \rightarrow \underset{j}{Min}\, x_j(T)$

geometric یا همگرایی نمایی: نرخ همگرایی $\xrightarrow{\text{سرعت همگرایی}}$ نمایی $a^T, a<1$ $>$ خطی $\frac{a}{T}$ $>$ غیرخطی $\frac{a}{\sqrt{T}}$

نکته: ماتریس $\Phi(T,0) = A(T)A(T-1)\dots A(0)$ هنگامی که $T\to\infty$ به یک ماتریس ثابت مانند $\bar{\Phi}(0)$ همگرا می‌شود.

(همه درایه‌های این ماتریس همگرا می‌شوند.)

$\lim_{T\to\infty} x(T) = 1\alpha \rightarrow$ اجماع — این مقدار α به شرایط اولیه بستگی دارد!

we know: $x(T+1) = \Phi(T,0)x(0) \Rightarrow$ if $T\to\infty$: $\boxed{1\alpha(x(0)) = \Phi(T,0)x(0)} \quad \forall x(0)$

همانطور که گفته شد، درایه‌های $\Phi(T,0)$ بعد از مدتی اکیداً مثبت (نامنفی بود) پس در نهایت به مقدار ثابت همگرا می‌شوند.

$\alpha(x(0)) = \Phi_i^T(T,0)\, x(0)$, $T\to\infty$ ، برای همه i ها — از شرایط اولیه مستقل می‌باشد.

اگر بعضی درایه‌های سطر $\lim_{T\to\infty} \Phi_i^T(T,0)$ به مقدار مشخصی همگرا نشوند (تغییر کنند)، در این صورت به ازای آن

شرایط اولیه غیرصفری مانند $x(0)$ وجود خواهد داشت که به ازای آن مقدار $\lim_{T\to\infty} x_i(T)$ همگرا نخواهد شد. در نتیجه

از اینجا که همگرایی همه $x_i(T)$ ها به مقدار اجماعی α اثبات شد، بنابراین خواهیم داشت: $\lim_{T\to\infty}\Phi(T,0) = \bar{\Phi}(0)$

ماتریس Φ همگرا خواهد شد!

Subject :

Year .   Month .   Date .

بدست آوردیم: $\lim_{T\to\infty} \Phi(T,0) = \bar{\Phi}(0) \Rightarrow 1\alpha = \bar{\Phi}(0)x(0)$

$$\begin{cases} \alpha = \bar{\Phi}_i^T(0)\,x(0) \\ \alpha = \bar{\Phi}_j^T(0)\,x(0) \end{cases} \Rightarrow \left(\bar{\Phi}_i^T(0) - \bar{\Phi}_j^T(0)\right)x(0) = 0 \quad \forall x(0) \in \mathbb{R}^m \Rightarrow \bar{\Phi}_i^T(0) = \bar{\Phi}_j^T(0) \quad \forall i,j$$

در نتیجه همه سطرهای ماتریس $\bar{\Phi}(0)$ با هم برابرند:

$$\bar{\Phi}(0) = \underset{m\times 1}{1}\ \underset{1\times m}{\pi^T} = \begin{bmatrix} \pi_1 & \cdots & \pi_m \\ \vdots & & \vdots \\ \pi_1 & \cdots & \pi_m \end{bmatrix}_{m\times m}$$

$$1\alpha = \bar{\Phi}(0)x(0) \Rightarrow \alpha = \pi^T x(0) = \sum_{i=1}^{m} \pi_i x_i(0) \quad \left(\sum_{i=1}^{m} \pi_i = 1 \ ,\ \pi_i > 0\right)$$

بنابراین تعادل اجماع یک convex combination از شرایط اولیه عامل‌ها خواهد بود

مرجع: A. Nedic, A. Ozdaglar: "Distributed subgradient Methods for Multi agent optimization"

ادامه: در مرجع فوق نشان داده شده است که نرخ همگرایی درایه‌های ماتریس Φ به صورت زیر است:

$$\left|[\Phi(k,s)]_{ij} - \pi_j(s)\right| \leqslant c\, q^{\frac{(k-s)}{B_0}} \rightarrow \begin{cases} k-s = \ell B_0 \\ q \in (0,1) \\ c > 0 \end{cases}$$

1) نشان می‌دهد که همگرایی ماتریس Φ هم به صورت نمایی می‌باشد!

2) به ازای همه jها ($agent$ها) درایه‌های سطر iام به π_j همگرا می‌شوند!

3) برابر شدن سطرها در نهایت به این معنی است که تاثیرپذیری agentهای مختلف از شرایط اولیه یکسان است

برای حالت خاص، ماتریس $A(k)$ تصادفی دوگانه باشد (جمع همه سطرها و جمع ستون‌های آن برابر یک باشد)

$A(k)1 = 1$ and $1^T A(k) = 1^T$

$$\frac{1}{m}\sum_{i=1}^{m} x_i(k+1) = \frac{1}{m} 1^T x(k+1) = \frac{1}{m} \underbrace{1^T A(k)}_{1^T} x(k) = \frac{1}{m} 1^T x(k) = \frac{1}{m}\sum_{i=1}^{m} x_i(k)$$

Subject :

Year . Month . Date .

در نتیجه مقدار اجماع همان مقدار متوسط شرایط اولیه خواهد بود. $\alpha = \frac{1}{m}\sum_{i=1}^{m} x_i(0) \rightarrow$ Average Consensus

$= \lim_{k\to\infty} \frac{1}{m}\sum_{i=1}^{m} x_i(k)$ or $\lim_{k\to\infty} x_i(k) = \alpha$

داشتیم ، $\bar{\Phi}(0) = 1\Pi^T = \begin{bmatrix} \Pi_1 & \cdots & \Pi_m \\ \Pi_1 & & \vdots \\ \vdots & & \\ \Pi_1 & \cdots & \Pi_m \end{bmatrix}$

نکته ۱: اگر A و B تصادفی دوبل باشند ضرب آنها هم تصادفی دوبل است:

اثبات

A و B تصادفی دوبل هستند

$AB1 = A1 = 1$

$1^TAB = 1^TB = 1^T$

بنابراین ماتریس $\bar{\Phi}(0)$ نیز تصادفی دوبل است (اگر شبکه طوری طراحی شده باشد که جمع وزن‌های خارج و وارد شونده به هر agent برابر یک باشد)

$\Rightarrow \sum_{i=1}^{m} \Pi_i = 1 \Rightarrow m\Pi_i = 1 \Rightarrow \Pi_i = \frac{1}{m} \quad i=1,2,\dots,m \Rightarrow \bar{\Phi}(0) = \frac{1}{m}11^T$

$\left| [\Phi(k,s)]_{ij} - \frac{1}{m} \right| \le C q^{\frac{(k-s)}{B_0}} \rightarrow \begin{cases} k-s = lB_0 \\ q \in (0,1) \\ C > 0 \end{cases}$

Consenus: Applications and Extensions

1) Opinion Dynamics in Social Network:

The French-DeGroot model of opinion formation: $\overbrace{x_i}^{\text{opinions}}(k+1) = \sum_{j=1}^{n} w_{ij}\, x_j(k), \ \forall i, \ k = 0,1,\dots$

Influence weights: is the contribution of agent j's opinion to the opinion of agent i at its next step.

The self-influence weight w_{ii} indicates the agent's openness to the assimilation of the other's opinion

$\left.\begin{array}{l} w_{ii} = 1 \equiv \text{Fully stubborn} \\ w_{ii} = 0 \equiv \text{Non-stubborn} \end{array}\right\} \rightarrow$ Opinions may not reach consensus but split into several clusters.

$0 < w_{ii} < 1 \equiv$ partially stubborn

مثال: 1——2

۱) اگر $w_{11} = 1$ و $w_{22} = 1$ باشد آنگاه مقدار نهایی هر Node همان مقدار اولیه آن خواهد بود

۲) اگر $w_{11} = 1$ و $w_{22} = 0$ باشد آنگاه مقدار نهایی هر دو Node مقدار اولیه Node 1 خواهد بود

Batus

Subject :
Year . Month . Date .

2) Hegselmann-Krause (HK) Model of opinion Formation:

Neighbors are defined in terms of a confidence interval: $N_i(t) = \{ j \in [m] \mid \| x_j(t) - x_i(t) \| \leq \epsilon \}$

Each agent updates his opinion by averaging the opinions of all his neighbors:

$$x_i(t+1) = \frac{1}{|N_i(t)|} \sum_{j \in N_i(t)} x_j(t) \quad \text{for } t = 0, 1, 2, \dots$$

Convergence: For any initial profile and any confidence interval ϵ, the agents opinions $x_i(t)$ converge to some limiting opinion x_i^* in a finite time, i.e, for each agent i, there exists time $T_i \geq 0$ such that: $x_i(t) = x_i^*$ for all $t \geq T_i$

The fragmentation of opinion may occur i.e, we may have agents i and j such that:

$$x_i^* \neq x_j^*$$

3) Multi-Vehicle formation control:

حرکت که اختلاف سرعت ربات ها با شده به ربات ها اعمال میشه

Set of variables / Robots

به مرور هم سرعت ربات ها برابر شود ← Dynamic & Continuous variable ← $\dot{x}_i(t) = u_i(t)$ $i = 1, 2, 3, \dots, n$ $\quad N = \{1, 2, \dots, n\}$

بجای بقیه method ها x در این روش $x_i(t)$ به صورت مستقیم تعیین نمی‌شود؛ بلکه با تغییر سرعت agent ها پیدا می‌کنیم

$$u_i = \sum_{\forall j \in N_i} \underbrace{a_{ij} (\overbrace{x_j - x_i}^{\text{اختلاف سرعت}})}_{\text{weighted avg}}$$

تصمیم میگیریم $u_i(t)$ چه باشد،

الگوریتم Consensus را نحوی پیدا می‌سازیم $u_i(t)$ مشخص می‌کند. هدف ما در این سناریو این است که تابع زیر را Min کنیم:

$$\Phi_G(x) = \frac{1}{2} \sum_{i,j=1}^{n} a_{ij} (\underbrace{x_j - x_i}_{\text{اختلاف سرعت agent i}})^2 \rightarrow$$ مجموع فاصله ها را مربعی حساب ≡ cost or loss

$$u = -\frac{1}{2} \nabla \Phi_G(x)$$

Replacing u_is in the local dynamics we have: $\dot{x}(t) = -L x(t)$ → Laplacian Matrix

لاپلاسین

$$l_{ij} = \begin{cases} \sum_{k=1, k \neq i}^{n} a_{ik}, & j = i \\ -a_{ij}, & j \neq i \end{cases}$$

یک روش مشخص برای ساخت L:

$L(G) = L = \Delta - A$ → weight matrix

$$\Delta = \text{diag}\left\{ \sum_{j=1}^{n} a_{ij} \right\}_{i=1}^{n}$$

Subject :

Year . Month . Date .

Every row sum of the Laplacian matrix is zero. Therefore, the Laplacian matrix always has a zero eigenvalue corresponding to a right eigenvector:

متناظر با مقدار ویژه L   بردار ویژه راست برای L

$L\mathbf{1} = 0 \Rightarrow \lambda_r = 0,\ v_r = \mathbf{1} = (1,1,\dots,1)^T \Rightarrow \mathrm{Rank}(L) \leq n-1$

بنابراین ماتریس L، full rank نخواهد بود؛ حال می‌خواهیم بررسی کنیم که در چه صورت rank این ماتریس دقیقاً n-1 خواهد بود.

Theorem 1: Let $G = (V, E, A)$ be a weighted digraph with Laplacian L. if G is strongly connected, then $\mathrm{rank}(L) = n-1$.

حال شرط strongly connected را اضافه می‌کنیم و به بررسی حالت‌های دیگر می‌پردازیم:

Convergence: The portocol $u_i = \sum_{v_j \in N_i} a_{ij}(x_j - x_i)$ globally asymptotically solves a consensus problem.

Sketch of proof:

1) $\mathrm{rank}(L) = n-1$

2) L has an eigenvalue at zero, the rest of eigenvalues have positive real parts → cause: $x^T L x = \frac{1}{2}\sum_{ij} a_{ij}(x_j - x_i)^2 \geq 0$

رتبه ماتریس n-1 است و طبق فرمول بالا semi-pos می‌باشد، پس بقیه مقادیر ویژه باید مثبت باشند!

3) Therefore the linear system is stable: $\dot{x}(t) = -Lx(t)$

تمام مقادیر ویژه این ماتریس (به جز صفر) منفی می‌باشند که باعث می‌شود این سیستم خطی پایدار باشد.

$\dot{x} = -Lx$

4) Equilibrium: $Lx^* = 0 \Rightarrow x^* \in \mathrm{Null}(L) \Rightarrow x^* = \alpha\mathbf{1}$

نقطه تعادل که در آن مشتق x (یعنی $\dot{x}$) صفر باشد را تعریف می‌کنیم.

پس x^* باید در فضای پوچی L باشد؛ رنک ماتریس n-1 است،

بعد فضای پوچی آن یک می‌باشد و متناظر با مقدار ویژه صفر

بردار ویژه 1 را داریم بنابراین:

Therefore:

بنابراین تمام x ها به هم رسیدند، پس اجماع داریم!

Batus

Subject :

Year . Month . Date .

حال می‌خواهیم این مبحث را تعمیم دهیم و به جای تعیین سرعت، به تعیین شتاب بپردازیم. ‌‌ ‌ ورودی شتاب!

4) Flocking

$$\begin{cases} \dot{q}_i = p_i \\ \dot{p}_i = u_i \end{cases}$$ ورودی ما

$q_i \in \mathbb{R}^m$ denote the position of node i

what we want: $\|q_j - q_i\| = d \quad \forall j \in N_i(q)$

یعنی می‌خواهیم فاصله هر agent از agent های همسایه‌اش برابر d باشد!

Variants/Extensions of Consensus:

1) Constrained Consensus:

$$x_i(t+1) = P_{X_i}\left[\sum_{j=1}^{m} A_{ij}(t)\, x_j(t)\right] \quad , \quad X \triangleq \bigcap_{i=1}^{m} X_i$$

$X \neq \emptyset$

Assumption: The sets $X_i \subseteq \mathbb{R}^n$ are nonempty, closed, and convex, and their interaction is nonempty.

2) Noisy Links:

$$x_i(k+1) = x_i(k) - \alpha_k \sum_{j \in N_i(k)} w_{ij}(k)\left(x_i(k) - x_j(k) - \zeta_{ij}(k)\right)$$

اثر نویز کانال بر اطلاعات ← اطلاعات j برای i به وزن w_{ij} ارسال می‌شود

خطا به آن می‌نشیند

The term $x_j(k) + \zeta_{ij}(k)$ represents a noisy estimates that agent i receives from its neighboring agent j, with $\zeta_{ij}(k)$ being the random noise on direct link (j,i) at time k

3) Delay:

$$x^i(k+1) = \sum_{j=1}^{m} a^i_j(k)\, x^j(k - t^i_j(k)) \quad \text{for } k = 0,1,2,\dots$$

The scalar $t^i_j(k)$ represents the delay of a massage from agent j to agent i.

4) Random Graph:

ارتباطات agent های ما به صورت تصادفی برقرار شود؛ یعنی نمی‌دانیم به طور deterministic داشته باشیم یا نه

از قبل مشخص نشده کدام agent به کدام agent اطلاعات ارسال می‌کند

set of stochastic matrices of order n with strictly positive diagonal entries

infinite dimensional vector of i.i.d

Distribution درایه

$\Omega = \{(w_1, w_2, \dots) \mid w_k \in \Omega_0\}$, $\Omega_i \sim w_i$ for $i = 1, 2, \dots$ $\quad \omega = (w_1, w_2, \dots) \in \Omega$ stochastic matrices with positive diagonals

$$x(k) = W_k(\omega)\, x(k-1) \ , \ W_k : \Omega \to \Omega_0 \ , \ W_k(\omega) = w_k$$

Batus

Subject :

Year . Month . Date .

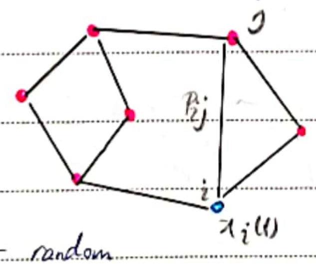
5) Asynchronous updating

clock

Drawbacks of synchronous updates:

of its clock

- Agents see the same time (Imagine that each has its own clock and it updates at each tick.)
- Precending algorithm requires that all nodes have synchronized clocks
- Links have to be perfectly activated to transmit and receive information
- Communication portocols (require receiving "ack-s")
- Communications can be costly



Asynchronous Gossip Scheme:

- Given a static undirected graph $G = ([m], \mathcal{E})$
- Each node may wake up with equal probability $(\frac{1}{m})$
- Upon waking up, a node chooses one of its neighbors at random
- It contacts its neighbor j with probability P_{ij}
- Upon agent i establishing connection with agent j, the agent "talk"
- They exchange their information (say at time t, i talks to j: i sends $x_i(t)$ to j and j sends $x_j(t)$ to i)
- They perform some local updates
- Upon completing the updates: Both agents go to "sleep"

The nodes I_k and J_k exchange their current values $x_i(k)$ and $x_j(k)$.

Both nodes update as follows: $x_{I_k}(k+1) = \gamma\, x_{I_k}(k) + (1-\gamma)\, x_{J_k}(k)$

$$x_{J_k}(k+1) = \gamma\, x_{J_k}(k) + (1-\gamma)\, x_{I_k}(k)$$

Distributed Optimization:

هدف کمینه کردن مجموع توابع هدف m عامل است که براساس یک توپولوژی متغیر با زمان (یا ثابت) با هم در ارتباط می‌باشند.

$$\min_{x} \sum_{i=1}^{m} f_i(x)$$

هدف عامل‌ها این است که به صورت همکارانه مسئله بهینه‌سازی مقید زیر را حل کنند:

$$x \in X = \bigcap_{i=1}^{m} X_i$$

که در آن $f_i : \mathbb{R}^n \to \mathbb{R}$ تابعی convex است و اطلاعات مخفی عامل i محسوب می‌شود.

Subject :

Year . Month . Date .

$X_i \in \mathbb{R}^n$ یک مجموعه بسته و convex ، مقید محلی عامل i ام می‌باشد و همچنین عامل‌ها به اطلاعات مجموعه مشترک $X = \bigcap_{i=1}^{m} X_i$ دسترسی دارند. مقدار بهینه مسئله فوق را با f^* نمایش می‌دهیم و مجموعه پاسخ‌های بهینه مسئله را با X^* نمایش می‌دهیم.

فرض می‌شود f_i ها لزوما در همه نقاط مشتق‌پذیر نیستند و بنابراین به جای آنها زیرگرادیان به صورت زیر تعریف می‌گردد:

$$f_i(\bar{x}) - f_i(x) \leq S_{f_i}(\bar{x})(x - \bar{x}) \quad \forall x$$

که در آن $S_{f_i}(\bar{x}) \in \mathbb{R}^n$ زیرگرادیان f_i در $\bar{x}$ می‌باشد:

$$S_{f_i}(\bar{x}) \in \partial f_i(\bar{x}) \subseteq \mathbb{R}^n$$

مجموعه زیرگرادیان تابع f در $\bar{x}$

الگوریتم تصویرسازی زیرگرادیان توزیع شده: روند الگوریتم به این صورت می‌باشد که هر عامل از یک تخمین اولیه از جواب مسئله بهینه‌سازی به صورت $x^i(0) \in X \subseteq \mathbb{R}^n$ شروع می‌کند و در هر مرحله از الگوریتم این تخمین را به‌روز می‌کند.

تخمین عامل i از x^*

این به‌روزرسانی شامل: ۱- ترکیب تخمین هر عامل با عامل‌های همسایه

۲- حرکت در جهت عکس زیرگرادیان جهت کمینه کردن f_i

۳- تصویرسازی مقدار به دست آمده بر مجموعه X است.

بدین ترتیب هر عامل i مراحل فوق را به شکل زیر انجام می‌دهد:

① شروع از شرط اولیه $x^i(0)$ ، $i = 1, \dots, m$ ، $k = 0$

② ترکیب اطلاعات $v^i(k) = \sum_{j=1}^{m} a^i_j(k) x^j(k)$

تصویرسازی

③ $x^i(k+1) = P_X[v^i(k) - \alpha_k d_i(k)]$

④ $k = k+1$

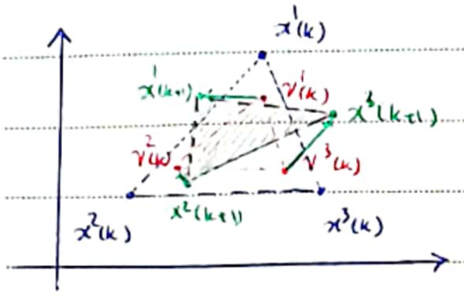
⑤ برو به مرحله ② تا شرط توقف

در رابطه فوق $a^i_j(k)$ ضرایب ماتریس وزنی $A(k)$ و $\alpha_k > 0$ گام به‌روزرسانی و $d_i(k)$ زیرگرادیان تابع $f_i(x)$ در $x = v^i(k)$ می‌باشد.

Subject :

Year . Month . Date .

مثال:

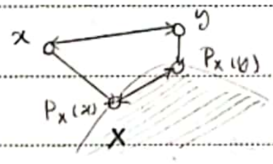


سوال اصلی: آیا بر اساس الگوریتم فوق، مقادیر تخمین عامل‌ها به مقدار بهینه مسئله اصلی همگرا می‌شوند؟ (اجماع + بهینگی)

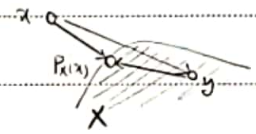
Convergence of Distributed Optimization Algorithm:

برخی خواص تصویرسازی:

$P_X(\bar{x}) = \underset{x\in X}{\mathrm{Argmin}}\ \|\bar{x}-x\| \rightarrow$ تصویرسازی



1) $\|P_X(x)-P_X(y)\| \leq \|x-y\| \rightarrow$ Non expansivity



2) $\|P_X(x)-y\|^2 \leq \|x-y\|^2 - \|P_X(x)-x\|^2 \quad \forall y\in X$

مفروضات:

1) Weight rule: if $a_i^j > 0$ then $a_i^j \geq \eta$, $a_i^i \geq \eta$

2) Doubly stochasticity of $A(k)$: $A(k) = [a_j^i(k)]_{ij}$ $\quad 1^T A(k) = 1^T$, $A(k)1 = 1$

3) $G(V, E_\infty)$: Strongly connected: $E_\infty = \{(j,i) \mid (j,i)\in E_k$ for infinitely many times $k\}$

often

اجتماع یال‌هایی که به صورت infinitely often ظاهر می‌شوند باید strongly connected باشد.

4) به ازای هر $(j,i)\in E_\infty$ ، $B\geq 1$ وجود دارد، که j حداکثر پس از B تکرار برای i اطلاعات خود را ارسال می‌کند.

3+4 = Periodically strongly connected

5) فرض می‌شود مجموعه شدنی مسئله بهینه‌سازی محدب و بسته و برای همه عامل‌ها معلوم است:

$\bigcap_{i=1}^{m} X_i = X$ (closed and convex)

6) فرض می‌شود گرادیان f_i در مجموعه X محدود است، به عبارت دیگر:

Batus $\|d\| \leq L > 0$, $\forall d\in \partial f_i(x)$ و $\forall x\in X$

الگوریتم بهینه‌سازی توزیع شده:

$$v^i(k) = \sum_{j=1}^{m} a_j^i(k)\, x^j(k)$$

$$x^i(k+1) = v^i(k) - \alpha_k d_i(k) + \varphi^i(k) \longrightarrow$$ اثر پروجکشن

$$\varphi^i(k) = P_X\left[v^i(k) - \alpha_k d_i(k)\right] - \left(v^i(k) - \alpha_k d_i(k)\right)$$

سوالات: • آیا این الگوریتم همچنان به اجماع عامل‌ها منجر می‌شود؟

• آیا این اجماع به صورت دقیق در نقطه بهینه تابع هدف خواهد بود؟

بررسی اجماع: از آنجا که ماتریس وزنی ماتریس به صورت تصادفی دوبل تعریف شده است. انتظار داریم عامل‌ها به مقدار متوسطشان همگرا شوند. بنابراین متغیر متوسط عامل را تعریف می‌کنیم:

$$y(k) = \frac{1}{m}\sum_{j=1}^{m} x^j(k)$$

$$y(k+1) = \frac{1}{m}\sum_{j=1}^{m} x^j(k+1) = \frac{1}{m}\sum_{i=1}^{m} v^i(k) - \frac{1}{m}\alpha_k \sum_{i=1}^{m} d_i(k) + \frac{1}{m}\sum_{i=1}^{m} \varphi^i(k)$$

$$\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{m} a_j^i x^j(k) = \frac{1}{m}\sum_{j=1}^{m} x^j(k) \sum_{i=1}^{m} a_j^i = y(k)$$

$$\Rightarrow y(k+1) = y(k) - \frac{\alpha_k}{m}\sum_{i=1}^{m} d_i(k) + \frac{1}{m}\sum_{i=1}^{m} \varphi^i(k)$$

قضیه: تحت فرضیات ذکر شده (۴-۱) اگر دنباله $\{x^j(k)\}$ توسط الگوریتم تصویرسازی زیرگرادیان توزیع شده (PDSG) تولید شود و دنباله $\{y(k)\}$ به شکل فوق تعریف شود آنگاه:

اجماع

1) $\lim_{k\to\infty} \|x^i(k) - y(k)\| = 0, \quad \forall i \text{ if } \lim_{k\to\infty} \alpha_k = 0$

محدود بودن مجموع وزن‌دار خطای اجماع در طول تکرارها

2) $\sum_{k=1}^{\infty} \alpha_k \|x^i(k) - y(k)\| < \infty \quad \forall i \text{ if } \sum_{k=1}^{\infty} \alpha_k^2 < \infty$

Subject :

Year . Month . Date .

اثبات (اجمالی):

رابطه برای x^i : $x(k+1) = \underbrace{A(k)x(k)}_{(I)} \underbrace{- \alpha_k d(k) + \varphi(k)}_{(II)}$

اصلی

$$\Phi(k,s) = A(k)A(k-1)\cdots A(s)$$

$$x(k+1) = \Phi(k,s)\,x(s) + [-\alpha_k d(k) + \varphi(k)] + \overbrace{\Phi(k,k)}^{A(k)}[-\alpha_{k-1} d(k-1) + \varphi(k-1)]$$

$$+ \underbrace{\Phi(k,k-1)}_{A(k)A(k-1)}[-\alpha_{k-2} d(k-2) + \varphi(k-2)] + \cdots + \Phi(k,s+1)[-\alpha_s d(s) + \varphi(s)]$$

بازنویسی رابطه بالا در حالت برداری برای عامل i ام: درایه‌های i ام سطر ماتریس Φ در بردارهای x^j و d_j و φ^j ضرب می‌شود.

$$x^i(k+1) = \boxed{\sum_{j=1}^{m} [\Phi(k,s)]^i_j\, x^j(s)} \xrightarrow{\frac{1}{m}\sum_{i=1}^{m}} y(k+1) = \boxed{y(s)} - \frac{1}{m}\boxed{\sum_{r=s}^{k-1}\sum_{j=1}^{m} \alpha_r d_j(r)}$$

$$- \boxed{\sum_{r=s}^{k-1}\sum_{j=1}^{m} [\Phi(k,r+1)]^i_j\, \alpha_r d_j(r)} - \boxed{\alpha_k d_i(k)} \qquad - \boxed{\frac{\alpha_k}{m}\sum_{i=1}^{m} d_i(k)} + \boxed{\frac{1}{m}\sum_{r=s}^{k-1}\sum_{j=1}^{m} \varphi^j(r)}$$

$$+ \boxed{\sum_{r=s}^{k-1}\sum_{j=1}^{m} [\Phi(k,r+1)]^i_j\, \varphi^j(r)} + \boxed{\varphi^i(k)} \qquad + \boxed{\frac{1}{m}\sum_{i=1}^{m} \varphi^i(k)}$$

اگر بخواهیم فاصله $x^i(k+1)$ و $y(k+1)$ را حساب کنیم، باید نرم ترم‌های متناظر را حساب کنیم و اینجا می‌توان از نامساوی مثلثی استفاده کرد!

یعنی جای همه k ها گذاشتیم

حال برای $s=0$ و $\boxed{k+1 \to k}$ اندازه تفاضل $x^i(k)$ و $y(k)$ را داریم:

$$\|x^i(k) - y(k)\| \le \sum_{j=1}^{m} \overbrace{\left|[\Phi(k-1,0)]^i_j - \frac{1}{m}\right|}\, \|x^j(0)\| \qquad y(0) = \frac{1}{m}\sum_{j=1}^{m} x^j(0)$$

$$+ \sum_{r=0}^{k-2}\sum_{j=1}^{m} \overbrace{\left|[\Phi(k-1,r+1)]^i_j - \frac{1}{m}\right|}\, \alpha_r \|\underbrace{d_j(r)}_{L}\|$$

استفاده از نامساوی مثلثی، نرم در جمع و ضرب.

$$+ \alpha_{k-1}\|\underbrace{d_i(k-1)}_{L}\| + \frac{\alpha_{k-1}}{m}\sum_{j=1}^{m}\|\underbrace{d_j(k-1)}_{L}\|$$

قبلا نشان دادیم که: $\left|[\Phi(k,s)]^i_j - \frac{1}{m}\right| \le C\beta^{k-s}$

$0 < \beta < 1$

$$+ \sum_{r=0}^{k-2}\sum_{j=1}^{m} \overbrace{\left|[\Phi(k-1,r+1)]^i_j - \frac{1}{m}\right|}\, \|\varphi^j(r)\|$$

$$+ \|\varphi^i(k-1)\| + \frac{1}{m}\sum_{i=1}^{m}\|\varphi^i(k-1)\|$$

Subject :

Year . Month . Date .

بنابراین: $\|x^i(k)-y(k)\| \le c\beta^{k-1}\sum_{j=1}^{m}\|x^j(0)\| + mcL\sum_{r=0}^{k-2}\beta^{k-r-2}\alpha_r$

$+2\alpha_{k-1}L + c\sum_{r=0}^{k-2}\beta^{k-r-2}\sum_{j=1}^{m}\|\varphi^j(r)\| + \|\varphi^i(k-1)\| + \frac{1}{m}\sum_{j=1}^{m}\|\varphi^j(k-1)\|$

نشان می‌دهیم در عبارت فوق خطای پروجکشن $\varphi^j(k)$ با ضریب α_k به سمت صفر می‌رود.

بر اساس خاصیت تصویرسازی: (شماره ۲) $\|P_X(x)-y\|^2 \le \|x-y\|^2 - \|P_X(x)-x\|^2 \quad \forall y\in X$

همچنین داریم: $\forall x^i \in \overset{\text{convex}}{X} \Rightarrow v^i(k)\in X$ convex combination of x^i's

خاصیت ۲: $\|\underbrace{x^i(k+1)}_{v^i(k)-\alpha_k d_i(k)+\varphi^i(k)} - v^i(k)\|^2 \le \|\overbrace{\cancel{v^i(k)} - \alpha_k \overset{L}{\cancel{d_i(k)}}}^{x} - \cancel{v^i(k)}\|^2 - \|\underbrace{x^i(k+1)-(v^i(k)-\alpha_k d_i(k))}_{\varphi^i(k)}\|^2$

$= \alpha_k^2L^2 - \|\varphi^i(k)\|^2$

می‌دانیم نرم یک مقدار نامنفی است پس:

$\Rightarrow 0 \le \alpha_k^2L^2 - \|\varphi^i(k)\|^2$

$\Rightarrow \|\varphi^i(k)\|^2 \le \alpha_k^2L^2 \Rightarrow \|\varphi^i(k)\| \le \alpha_k L$

★ با جایگذاری $\alpha_k L$ به جای φ^i در نامساوی داریم: $\|x^i(k)-y(k)\| \le c\beta^{k-1}\sum_{j=1}^{m}\|x^j(0)\| + 2mcL\sum_{r=0}^{k-2}\beta^{k-r-2}\alpha_r + 4\alpha_{k-1}L$

در بی‌نهایت

با توجه به این موضوع صفر می‌شوند

در حالت حدی: $0<\beta<1 \Rightarrow \beta^k \to 0 \;,\; \alpha_k \to 0$

$\Rightarrow \lim_{k\to\infty}\|x^i(k)-y(k)\| \le 2mcL\beta^{-2}\lim_{k\to\infty}\sum_{r=0}^{k-2}\beta^{k-r}\alpha_r$

حال می‌خواهیم این حد را بررسی کنیم!

$$\sum_{r=0}^{k}\beta^{k-r}\alpha_r \le \underbrace{\gamma_K}_{\max\limits_{0\le r\le K}\alpha_r}\sum_{r=0}^{K}\beta^{k-r} + \underbrace{\epsilon_k}_{\max\limits_{K+1\le r\le k}\alpha_r}\sum_{r=K+1}^{k}\beta^{k-r} \le \gamma_K\beta^k\frac{1-\beta^{-(K+1)}}{1-\beta^{-1}} + \epsilon_k\frac{1}{1-\beta}$$

Subject :

Year . Month . Date .

اگر $K=\frac{k}{2}$ باشد و $k\to\infty$ میل کند آنگاه K نیز به بی‌نهایت میل می‌کند. حال می‌خواهیم ببینیم عبارت فوق چه می‌شود:

سپس داریم:

$$\to \beta^{k}\frac{1-\beta^{-(K+1)}}{1-\beta^{-1}}=\frac{\beta^{k}-\beta^{\frac{k}{2}-1}}{1-\beta^{-1}}\xrightarrow{\text{if } k\to\infty}\beta^{k}\to 0,\ \beta^{k/2-1}\to 0$$

$$\Rightarrow \frac{\beta^{k}-\beta^{k/2-1}}{1-\beta^{-1}}\to 0$$

$$\varepsilon_k\frac{1}{1-\beta}\xrightarrow{\text{if } k\to\infty}\varepsilon_k\to 0$$

همچنین چون $\varepsilon_k=\max\limits_{K+1\le r\le k}\alpha_r$ و چون k و K هر دو به بی‌نهایت میل می‌کنند پس ε_k به صفر میل می‌کند پس کل عبارت به صفر میل می‌کند

$$\text{Therefore: } \lim_{k\to\infty}\|x^{i}(k)-y(k)\|=0\quad \forall i$$

اثبات (محدود بودن مجموع فاصله از حد اجماع): طرفین رابطه ★ را در α_k ضرب می‌کنیم:

$$\alpha_k\|x^{i}(k)-y(k)\|\le C\alpha_k\beta^{k-1}\sum_{j=1}^{m}\|x^{j}(0)\|+2mcL\sum_{r=0}^{k-2}\beta^{k-r-2}\alpha_k\alpha_r+4\alpha_k\alpha_{k-1}L$$

حال از نامساوی‌های کمکی زیر جهت مربع کامل سازی ضرایب استفاده می‌کنیم:

$$\alpha_k\beta^{k-1}\le\alpha_k^{2}+\beta^{2(k-1)}$$

$$2\alpha_k\alpha_r\le\alpha_k^{2}+\alpha_r^{2}$$

$$\Rightarrow \alpha_k\|x^{i}(k)-y(k)\|\le C\beta^{2(k-1)}\sum_{j=1}^{m}\|x^{j}(0)\|+C\alpha_k^{2}\sum_{j=1}^{m}\|x^{j}(0)\|$$

از $mC\alpha_k^2$ فاکتور می‌گیریم ←

$$+mC\alpha_k^{2}L\underbrace{\sum_{r=0}^{k-2}\beta^{k-r-2}}+mcL\sum_{r=0}^{k-2}\beta^{k-r-2}\alpha_r^{2}+2L(\alpha_k^{2}+\alpha_{k-1}^{2})$$

$\le\frac{1}{1-\beta}$: اگر k را به بی‌نهایت ببریم (حد بگیریم) طبق سری هندسی این عبارت

$$\Rightarrow \alpha_k\|x^{i}(k)-y(k)\|\le C\beta^{2(k-1)}\sum_{j=1}^{m}\|x^{j}(0)\|+mcL\sum_{r=0}^{k-2}\beta^{k-r-2}\alpha_r^{2}+2L(\alpha_k^{2}+\alpha_{k-1}^{2})+mCA\alpha_k^{2}$$

$$A=\frac{1}{m}\sum_{j=1}^{m}\|x^{j}(0)\|+\frac{L}{1-\beta}$$

اگر از طرفین $\sum_{k=1}^{\infty}$ سیگما بگیریم:

$$\sum_{k=1}^{\infty}\alpha_k\|x^{i}(k)-y(k)\|\le\underbrace{C\sum_{k=1}^{\infty}\beta^{2(k-1)}\sum_{j=1}^{m}\|x^{j}(0)\|}_{T_1}+\underbrace{\sum_{k=1}^{\infty}(mCA\alpha_k^{2}+2L(\alpha_k^{2}+\alpha_{k-1}^{2}))}_{T_2}$$

$$+\underbrace{mcL\sum_{k=1}^{\infty}\sum_{r=0}^{k-2}\beta^{k-r-2}\alpha_r^{2}}_{T_3}$$

T_1 محدود است چرا که $\sum_{k=1}^{\infty} \beta^{2(k-1)}$ محدود است.

T_2 محدود است چرا که $\sum_{k=1}^{\infty} \alpha_k^2 < \infty$ محدود است.

برای T_3 داریم:

$$\sum_{k=1}^{\infty} \sum_{r=0}^{k} \beta^{k-r} \alpha_r^2 = \sum_{r=0}^{k} (\alpha_r)^2 \sum_{k=r}^{\infty} \beta^{k-r} \leq \frac{1}{1-\beta} \underbrace{\sum_{r=0}^{k} (\alpha_r)^2}_{\text{if } k\to\infty :\ <\infty} < \infty$$

در نهایت نتیجه می‌شود:

$$\sum_{k=1}^{\infty} \alpha_k \| x^i(k) - y(k) \| < \infty$$

قضیه Optimality: اگر فرضیات بیان شده ۴-۱ برقرار باشند و $\{x^i(k)\}$ توسط معادلات زیر تولید شود:

$$v^i(k) = \sum_{j=1}^{m} a_j^i(k)\, x^j(k) \quad , \quad x^i(k+1) = P_X[v^i(k) - \alpha_k d_i(k)]$$

مجموعه پاسخ های بهینه

به طوریکه $\sum \alpha_k = \infty$ ، $\sum \alpha_k^2 < \infty$ باشد و همچنین مجموعه X^* غیرتهی باشد. در این صورت پاسخ بهینه $x^* \in X^*$

وجود دارد به طوریکه تخمین همه عامل ها به این پاسخ همگرا می‌شود به عبارت دیگر:

$$\lim_{k\to\infty} \| x^i(k) - x^* \| = 0 \quad \forall i$$

اثبات: اگر نشان دهیم که عبارت زیر نزولی می‌باشد (به ازای افزایش k) آنگاه قضیه اثبات می‌شود:

$$\sum_{i=1}^{m} \| x^i(k+1) - x^* \|^2 = \sum_{i=1}^{m} \| P_X[v^i(k) - \alpha_k d_i(k)] - x^* \|^2$$

خاصیت اپراتور تصویرسازی:

$$\| P_X(x) - P_X(y) \| \leq \| x - y \| \Rightarrow \| P_X[v^i(k) - \alpha_k d_i(k)] - \overbrace{P_X(x^*)}^{x^*} \| \leq \| v^i(k) - \alpha_k d_i(k) - x^* \|$$

$$\Rightarrow \sum_{i=1}^{m} \| x^i(k+1) - x^* \|^2 \leq \sum_{i=1}^{m} \| v^i(k) - \alpha_k d_i(k) - x^* \|^2 = \sum_{i=1}^{m} (v^i(k) - \alpha_k d_i(k) - x^*)^T (v^i(k) - \alpha_k d_i(k) - x^*)$$

$$= \sum_{i=1}^{m} \| v^i(k) - x^* \|^2 + \sum_{i=1}^{m} \alpha_k^2 \| \cancel{d_i(k)} \|^2_{L} - 2\alpha_k \sum_{i=1}^{m} \underline{d_i^T(k)(v^i(k) - x^*)}$$

مبتنی بر تعریف subgradient:

$$d_i^T(k)(v^i(k) - x^*) \geq f_i(v^i(k)) - f_i(x^*)$$

Subject :

Year . Month . Date .

$$\Rightarrow \leq \sum_{i=1}^{m} \|v^i(k) - x^*\|^2 + m\alpha_k^2 L^2 - 2\alpha_k \sum_{i=1}^{m} [f_i(v^i(k)) - f_i(x^*)]$$

$$v^i(k) = \sum_{j=1}^{m} a_j^i x^j(k) \qquad x^* = \sum_{j=1}^{m} a_j^i x^*(k)$$

$$\sum_{i=1}^{m} \|\sum_{j=1}^{m} a_j^i(k)(x^j(k) - x^*)\|^2 \leq \sum_{i=1}^{m}\sum_{j=1}^{m} a_j^i(k) \|x^j(k) - x^*\|^2 = \sum_{j=1}^{m} \|x^j(k) - x^*\|^2$$

$$\Rightarrow \sum_{i=1}^{m} \|x^j(k+1) - x^*\|^2 \leq \sum_{j=1}^{m} \|x^j(k) - x^*\|^2 + m\alpha_k^2 L^2 - 2\alpha_k \sum_{i=1}^{m} [f_i(v^i(k)) - f_i(x^*) + f_i(y(k)) - f_i(y(k))]$$

$$\sum_{i=1}^{m} |f_i(v^i(k)) - f_i(y(k))| \leq \sum_{i=1}^{m} L \|\underbrace{v^i(k)}_{\sum_{j=1}^{m} a_j^i x^j} - \underbrace{y(k)}_{\sum_{j=1}^{m} a_j^i y}\| \leq L \sum_{j=1}^{m}\sum_{i=1}^{m} a_j^i \|x^i(k) - y(k)\| = \sum_{j=1}^{m} L \|x^j(k) - y(k)\|$$

بنابراین خواهیم داشت:

$$\sum_{i=1}^{m} \|x^j(k+1) - x^*\|^2 \leq \sum_{j=1}^{m} \|x^j(k) - x^*\|^2 + m\alpha_k^2 L^2 - 2\alpha_k \sum_{i=1}^{M} [f_i(y(k)) - f_i(x^*)]$$

$$+ 2\alpha_k L \sum_{i=1}^{m} \|x^i(k) - y(k)\|$$

در ادامه: از طرفین از زمان K تا N، $\sum_{k=K}^{N}$ می‌گیریم و فرض $f = \sum_{i=1}^{m} f_i$

$$\xrightarrow{\text{با ساده سازی}} \sum_{i=1}^{m} \|x^i(N+1) - x^*\|^2 \leq \sum_{i=1}^{m} \|x^i(K) - x^*\|^2 + mL^2 \sum_{k=K}^{N} \alpha_k^2 - 2\sum_{k=K}^{N} \alpha_k (f(y(k)) - f(x^*))$$

$$+ 2L \sum_{k=K}^{N} \alpha_k \sum_{i=1}^{m} \|x^i(k) - y(k)\|$$

با قرار دادن $K=0$ و $N \to \infty$ خواهیم داشت:

$$0 \leq \sum_{i=1}^{m} \|x^i(N+1) - x^*\|^2 \leq \underbrace{\sum_{i=1}^{m} \|x^i(0) - x^*\|^2}_{<\infty} + \underbrace{mL^2 \sum_{k=0}^{\infty} \alpha_k^2}_{<\infty \text{ فرض}} - 2\sum_{k=0}^{\infty} \alpha_k (f(y(k)) - f(x^*))$$

$$+ \underbrace{2L \sum_{i=1}^{m}\sum_{k=0}^{\infty} \alpha_k \|x^i(k) - y(k)\|}_{<\infty \text{ نشان دادیم}}$$

باید $<\infty$ باشد

چون اگر بی‌نهایت شود ...

Balus

Subject :
Year . Month . Date .

حال می‌خواهیم ببینیم به محدود بودن $\sum_{k=0}^{\infty} \alpha_k(f(y(k))-f(x^*))$ به چه دردی می‌خورد؟

fact: $\sum_{k=0}^{\infty} \alpha_k (f(y(k))-f(x^*)) < \infty$, $\sum_{k=1}^{\infty} \alpha_k = \infty$, $y(k) \in X \rightarrow f(y(k)) - f^* \geq 0$

اگر $\liminf_{k\to\infty} [f(y(k))-f^*]$ صفر نباشد، آنگاه $\sum \alpha_k (f(y(k))-f(x^*))$ به بی‌نهایت میل می‌کند، پس:

$\liminf_{k\to\infty} [f(y(k))-f^*]=0$

حال می‌خواهیم با میل دادن K و N به بی‌نهایت عبارت زیر را بررسی کنیم:

$$\sum_{i=1}^{m} \|x^i(N+1)-x^*\|^2 \leq \sum_{i=1}^{m} \|x^i(K)-x^*\|^2 + mL^2 \sum_{k=K}^{N} \cancel{\alpha_k^2}_{0} - 2 \sum_{k=K}^{N} \cancel{\alpha_k}_{0} (f(y(k))-f(x^*))$$
$$+2L \sum_{k=K}^{N} \cancel{\alpha_k}_{0} \sum_{i=1}^{m} \|x^i(k)-y(k)\|$$

در واقع چون K و N هر دو به بی‌نهایت میل می‌کنند و می‌دانیم α_k در بی‌نهایت صفر است، پس:

اگر K در بی‌نهایت غیر صعودی می‌شود و $\lim_{N\to\infty} \sum_{i=1}^{m} \|x^i(N+1)-x^*\|^2 \leq \lim_{K\to\infty} \sum_{i=1}^{m} \|x^i(K)-x^*\|^2$

(اما این محاسبه کار نداشت)

به ازای هر $\tilde{x} \in X^*$ دنباله $\|\tilde{x} - x^*\|$ و $\sum_{i=1}^{m} \|x^i(k)$ همگرا است. داشتیم $\lim_{k\to\infty} \|x^i(k)-y(k)\|=0$، بنابراین

دنباله $\{\|y(k)-\tilde{x}\|\}$ به ازای هر $\tilde{x}$ همگرا است. در نتیجه $y(k)$ به نقطه‌ای همگرا خواهد شد. از این به علاوه

پیوستگی تابع f و $\liminf_{k\to\infty} f(y(k))=f^*$ اثبات می‌شود $y(k)$ به یک $\tilde{x} \in X$ همگرا می‌شود.

در نهایت $\lim_{k\to\infty} y(k)=x^*$ و در نتیجه: $\lim_{k\to\infty} x^i(k)=x^* \quad \forall i$

Subject :

Year . Month . Date .

مسئله بهینه‌سازی توزیع‌شده با قیود ساده‌ی دوگانی : انگیزه ما برای مطرح کردن این بحث این است که projection در صورتی که قیود پیچیده باشند ممکن است باعث ایجاد چالش شود.

تعریف مسئله :

$$\text{Min} \sum_{i=1}^{N} f_i(x)$$

Convex $f_i : R^n \to R$ (private info of agent i)

X : Compact and Convex

$$\text{s.t. } g(x) \leq 0 \;, h(x) = 0 \;; x \in X$$

x : global decision variable

نامساوی‌ها در قیود مشترک، X تعریف شده است $g : R^n \to R^m$ Convex $\qquad g(x)$ is a vector $\Rightarrow g_l \leq 0 \;\; l \in \{1, \dots, m\}$

$h : R^n \to R^v \qquad h(x) = Ax - b \;, \; A \in R^{v \times n}$

مسئله فوق را به صورت زیر بازنویسی می‌کنیم :

$$\text{Min} \sum_{i=1}^{N} f_i(x)$$

چون تابع هدف مسئله جمع‌پذیر است [illegible] decompose

$$\text{s.t } N g(x) \leq 0 \;, \; N h(x) = 0 \;; x \in X$$

برای مسئله فوق تابع penalty dual را تعریف کرده و آن را بهینه می‌کنیم :

ابتدا تابع پنالتی را به صورت زیر تعریف می‌کنیم :

$$H : R^n \times R^m_{\geq 0} \times R^v_{\geq 0}$$

$$H(x, \mu, \vartheta) = f(x) + N\mu^T \overbrace{[g(x)]^+}^{\text{Max}(0, g(x)) \; (\text{II})} + N\vartheta^T \underbrace{|h(x)|}_{(\text{I})}$$

(I) و (II) عبارت تنبیهی ناشی از تخطی می‌باشند

بر این اساس تابع penalty dual به صورت زیر تعریف می‌شود :

$$q_p : R^m_{\geq 0} \times R^v_{\geq 0} \to R$$

$$q_p = \inf_{x \in X} H(x, \mu, \vartheta)$$

حال مسئله penalty dual به صورت زیر تعریف می‌شود :

$$\underset{\substack{\mu \in R^m_{\geq 0} \\ \vartheta \in R^v_{\geq 0}}}{\text{Max}} \; q_p(\mu, \vartheta) \qquad \text{s.t } \; \mu \geq 0 \;, \; \vartheta \geq 0$$

حال می‌خواهیم ببینیم H چه ویژگی‌هایی دارد و برای آن نشان دادیم : ([illegible])

Batus

Subject :

Year . Month . Date .

۱) فرض کنید (λ^*, μ^*) یک پاسخ بهینه مسئله dual باشد: $\mu^* \geq 0$

$$d_a^* = q_a(\mu^*, \lambda^*) = \inf_{x \in X} \{ f(x) + N\mu^{*T} g(x) + N\lambda^{*T} h(x) \} \quad (I)$$

$$\leq \inf_{x \in X} \{ f(x) + N\mu^{*T} [g(x)]^+ + N|\lambda^*|^T |h(x)| \}$$

مقدار بهینه تابع dual

$$= q_p(\mu^*, |\lambda^*|) \leq \max_{\mu \geq 0, \lambda \geq 0} q_p(\mu, \lambda) = d_p^* \quad (II)$$

$(I), (II) \Rightarrow d_a^* \leq d_p^*$ (1) مقدار بهینه تابع dual پنالتی

می‌دانیم به ازای هر $x^* \in X^*$ که feasible باشد داریم:

$x^* \in X^*$, $[g(x^*)]^+ = 0$, $|h(x^*)| = 0$

بنابراین خواهیم داشت:

$$q_p(\mu, \lambda) = \inf_x H(x, \mu, \lambda) \leq \underset{x^* \in X^*}{H(x^*, \mu, \lambda)} = f(x^*) = p^*$$

$$\Rightarrow \max_{\mu \geq 0, \lambda \geq 0} q_p(\mu, \lambda) = d_p^* \leq p^* \quad (2)$$

$(1), (2) \Rightarrow d_a^* \leq d_p^* \leq p^*$

← از نامساوی فوق روشن است اگر شرایطی برقرار باشد که $d_a^* = p^*$ باشد (مثلاً اگر مسئله بهینه‌سازی محدب و شرایط slater برقرار باشد) در این صورت خواهیم داشت: $d_p^* = p^*$

← duality gap برای مسئله dual penalty کوچکتر یا مساوی duality gap در مسئله dual است:
$p^* - d_p^* \leq p^* - d_a^*$

۲) مزیت دیگر تابع پنالتی نسبت به تابع لاگرانژ محدب بودن تابع پنالتی نسبت به متغیر x می‌باشد که در تابع لاگرانژ به دلیل امکان منفی بودن ضرایب لاگرانژ این تحقق نمی‌شود.

Batus

Subject :

Year . Month . Date .

قضیه نقطه زینی برای تابع پنالتی: چندتایی مرتب (x^*, μ^*, λ^*) یک نقطه زینی تابع پنالتی H بر $X \times R^{m}_{\geq 0} \times R^{v}_{\geq 0}$

می‌باشد اگر و فقط اگر این چندتایی مرتب مقادیر بهینه مسائل Primal و Penalty dual باشند و تساوی زیر برقرار باشد:

$$\underbrace{\sup_{\substack{\mu \geq 0 \\ \lambda \geq 0}} \inf_{x \in X} H(x, \mu, \lambda)}_{\text{Penalty dual حل مسئله}} = \underbrace{\inf_{x \in X} \sup_{\substack{\mu \geq 0 \\ \lambda \geq 0}} H(x, \mu, \lambda)}_{\text{Primal حل مسئله}}$$

به سادگی می‌توان نشان داد تابع پنالتی $H(x, \mu, \lambda)$ به صورت یک تابع Convex-concave می‌باشد:

$$H_i(x, \mu, \lambda) = f_i(x) + \mu^T [g(x)]^+ + \lambda^T |h(x)| \Rightarrow H(x, \mu, \lambda) = \sum_{i=1}^{N} H_i(x, \mu, \lambda)$$

$\to H_i(x, w)$: $w = (\mu, \lambda)$ fixed $\to$ Convex in x

$\to H_i(x, \mu, \lambda)$: x is fixed $\to$ Concave in (μ, λ)

با توجه به این نکته می‌توانیم از الگوریتم‌های زیرگرادیان در سوی گرادیان برای به دست آوردن نقطه زینی استفاده کنیم و در عین حال

چون این توابع به صورت جمع یک سری از توابع هستند، می‌توانیم از الگوریتم‌های توزیع شده استفاده کنیم.

الگوریتم توزیع شده زیرگرادیان برای حل مسئله Penalty primal dual: $\forall i$

(1) Initialization: $\lambda^i(0) \in R^v_{\geq 0}$, $\mu^i(0) \in R^m_{\geq 0}$, $x^i(0) \in X$, $y^i(1) = N f_i(x^i(0)) \rightarrow$ تخمین

(2) Information fusion: $v_x^i(k) = \sum_{j=1}^{N} a_j^i(k) x^j(k)$ فقط برای همسایگان عامل i در ماتریس A غیر صفر است

$$v_\mu^i(k) = \sum_{j=1}^{N} a_j^i(k) \mu^j(k)$$

$$v_\lambda^i(k) = \sum_{j=1}^{N} a_j^i(k) \lambda^j(k)$$

$$v_y^i(k) = \sum_{j=1}^{N} a_j^i(k) y^j(k)$$

Subject :

Year . Month . Date .

step size

(3) Optimization: $x^i(k+1) = P_X\left[v_x^i(k) - \alpha(k) D_x^i(k)\right]$

زیرگرادیان تابع $H_i(x, v_\mu^i(k), v_\lambda^i(k))$ نسبت به x در $x = v_x^i(k)$

$$D_x^i(k) = Df_i(v_x^i(k)) + \sum_{l=1}^{m} [v_\mu^i(k)]_l \, D[g_l(v_x^i(k))]^+ + \sum_{l=1}^{v} [v_\lambda^i(k)]_l \, D|h_l(v_x^i(k))|$$

← در عمل از سازوکاری با استفاده از Projection نیست تمام قیدها را می‌توان در $g(x)$ و $h(x)$ جا داد و فضای

feasible هم فضایی است که تمام شرایط آن برقرار خواهد بود.

$\mu^i(k+1) = v_\mu^i(k) + \alpha(k)\left[g(v_x^i(k))\right]^+ \rightarrow$ supergradient of H_i w.r.t μ at $(v_x^i(k), v_\mu^i(k), v_\lambda^i(k))$

$\lambda^i(k+1) = v_\lambda^i(k) + \alpha(k)\left[|h(v_x^i(k))|\right] \rightarrow$ supergradient of H_i w.r.t λ at "

← supergradient همان subgradient برای تابع مقعر می‌باشد.

$y^i(k+1) = v_y^i(k) + N\left(f_i(x^i(k)) - f_i(x^i(k-1))\right) \rightarrow$ هیچ تاثیری در روند بهینه‌سازی ندارد و صرفا پیشنهادی
برای پی بردن به همگرایی می‌باشد.

شرایط همگرایی برای $\alpha(k)$: $S(k) = \sum_{l=0}^{k} \alpha(l)$

1) $\lim_{k\to\infty} \alpha(k) = 0$ 2) $\lim_{k\to\infty} \alpha(k+1) S(k) = 0$

3) $\sum_{k=0}^{\infty} \alpha(k) = \infty$ 4) $\sum_{k=0}^{\infty} \alpha^2(k) < \infty$ 5) $\sum_{k=0}^{\infty} \alpha^2(k+1) S(k) < \infty$ 6) $\sum_{k=0}^{\infty} \alpha(k+1) S^2(k) < \infty$

Ex. $\alpha(k) = \frac{1}{k+1}$

نتیجه: همگرایی الگوریتم توزیع شده penalty - primal - dual مسئله بهینه‌سازی چندعامله مقید را در نظر

بگیرید و فرض کنید فرض‌های ① - ⑥ (که در بالا تعریف شده و تضمین شده بیان شد) برقرار باشد.

Subject :

Year . Month . Date .

Distributed — Penalty — Primal — Dual — Subgradient

در این صورت اگر $\{x^i(k)\}$ و $\{y^i(k)\}$ توسط الگوریتم DPPDS تولید شده باشند و step size $\alpha(k)$

شرایط فوق را برآورده کنند، آنگاه پاسخ $\tilde{x} \in X^*$ وجود دارد به طوری که:

$$\lim_{k\to\infty} \|x^i(k) - \tilde{x}\| = 0 \quad \forall i \in V$$

$$\lim_{k\to\infty} \|y^i(k) - p^*\| = 0 \quad \forall i \in V$$

بهینه $\sum f_i$

Distributed Optimization Extension and Applications:

Stochastic consensus-based subgradient methods:

$$x_i(t+1) = P_X\left[w_i(t+1) - \alpha(t)\left(\tilde{\nabla} f_i(w_i(t+1)) + e_i(t+1)\right)\right]$$

- Stochastic error in computing subgradient ∇f_i, measurement etc.
- Uniformly bounded in mean square
- Provide an estimate for the algorithm's performance in mean

Consensus-based subgradient method for noisy links:

$$w_i(t+1) = x_i(t) - \eta(t) \sum_{j=1} r_{ij}\left(x_j(t) + \varepsilon_{ij}(t)\right)$$

Noise: random zero mean noise on the link from j to i.

$\eta(t) > 0$ is a noise-damping stepsize

$r_{ij} = 0$ when $j \notin N_i$

$$x_i(t+1) = w_i(t+1) - \alpha(t) g_i(w_i(t+1))$$

- Noise-damping stepsize $\eta(t)$ has to be coordinated with sub-gradient related stepsize $\alpha(t)$:

$$\sum_t \alpha(t) = \infty, \quad \sum_t \alpha^2(t) < \infty$$

$$\sum_t \eta(t) = \infty, \quad \sum_t \eta^2(t) < \infty$$

$$\sum_t \alpha(t)\eta(t) < \infty, \quad \sum_t \frac{\alpha^2(t)}{\eta(t)} < \infty$$

Consensus-based subgradient methods for distributed sets:

- The set X can be given as intersection set $X = \bigcap_{i=1}^{m} X_i$ where each X_i is a private information of agent i.
- In this case the projection-based update of algorithm is modified as follows:

$$x_i(t+1) = P_{X_i}\left[w_i(t+1) - \alpha(t)\tilde{\nabla} f_i(w_i(t+1))\right]$$

Batus

Subject :

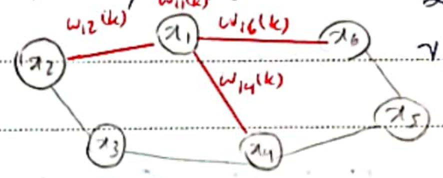
Year . Month . Date .

Distributed Random Projection (DRP) Algorithm:

A version of the algorithm where X_i is itself an intersection of a collection of sets, and a random set from the collection is used in the projection-based update.

Initialize $x_i(0)$ for $i \in V$. For $k \geq 0$, each agent i performs the following steps:

1. Mixing: $v_i(k) = \sum_{j=1}^{m} a_{ij}(k)\, x_j(k)$



$v_1(k) = w_{11}(k)\, x_1(k) + w_{12}(k)\, x_2(k) + w_{14}(k)\, x_4(k) + x_{16}(k)\, w_{16}(k)$

2. Gradient update: $\tilde{v}_i(k) = v_i(k) - \alpha_k \nabla f_i(v_i(k))$

3. Projection: A random variable $\Omega_i(k) \in I_i$ is drawn, and a component $X_i^{\Omega_i(k)}$ of $X_i = \bigcap_{j = I_i} X_i^j$ is used for projection: $x_i(k+1) = \Pi_{X_i^{\Omega_i(k)}} [\tilde{v}_i(k)]$

یک زیرمجموعه‌ای از ست

I_i مجموعه اندیس قیود agent i ام می‌باشد و $\Omega_i(k)$ یکی از زیرمجموعه‌های آن می‌باشد

Distributed Mini-Batch Random Projection (DMRP)

- What if X_i consists of 10^4 hyperplanes?
- Then one sample will render a poor approximation of the true set X_i.
- 100 samples will provide a better approximation of the set.

Initialize $x_i(0)$ for $i \in V$. For $k \geq 0$, each agent i does the following:

1. Mixing: $v_i(k) = \sum_{j=1}^{m} a_{ij}(k)\, x_j(k)$

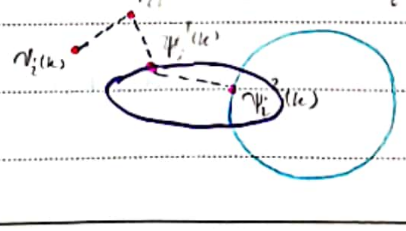
2. Gradient update: $\psi_i^0(k) = v_i(k) - \alpha_k \nabla f_i(v_i(k))$

3. Projection: A batch of b independent random variables $\Omega_i^r(k) \in J_i$, $r = 1, \dots, b$ is drawn. The components $X_i^{\Omega_i^1(k)}, \dots, X_i^{\Omega_i^b(k)}$ of $X_i = \bigcap_{j \in J_i} X_i^j$ are used for sequential projections.

$$\psi_i^r(k) = \Pi_{X_i^{\Omega_i^r(k)}} [\psi_i^{r-1}(k)], \quad \text{for } r = 1, \dots, b$$

$$x_i(k+1) = \psi_i^b(k)$$

$\psi_i^0(k) = v_i(k) - \alpha_k \nabla f_i(v_i(k))$



Batus

Subject :

Year . Month . Date .

Asynchronous consensus-based subgradient methods:

$$v_i(k) = \left(x_{I_k}(k-1) + x_{J_k}(k-1)\right)/2 \qquad i \in \{I_k, J_k\}$$

- Gossips

$$x_i(k) = \Pi_{X_i^{x_i(k)}} \left[v_i(k) - \alpha_i(k) \nabla f_i(v_i(k)) \right]$$

$$x_i(k) = x_i(k-1) \qquad \text{for all } i \notin \{I_k, J_k\}$$

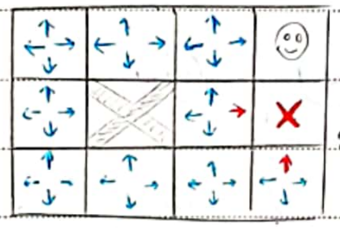
۸. تصمیمی که در State A از State بعدی خود تأثیر می‌پذیرد!

Dynamic Optimization

"Transition from optimization to RL"

Optimization in a Grid World:

برای آشنایی با RL با یک مثال شروع می‌کنیم:



هدف این است که در خانه‌هایی که با شیء مشخص شده حرکت کنیم و به (☺) برسیم:

هر حرکت یک cost دارد و در صورت رسیدن به ☺ یک award می‌گیریم. ما به دنبال

مسیری که بیشترین award-cost را داشته باشد؛ از هر خانه‌ای که باشیم به دنبال مسیر بهینه به هدف هستیم:

- We do not have an explicit model of cost function to minimize.
- We have a Multi-Stage decision making problem (otherwise we face curse of dimentionality)

ایده‌ای که اینجا مطرح می‌شود این است که یک مفهوم جدید به اسم state (حالت یا وضعیت) را در نظر بگیریم و این مسئله را به صورت مرحله‌ای!

و این تبدیل می‌شود به: در هر وضعیتی که هستیم چه تصمیمی بگیریم؟ به عبارت دیگر مسئله به یک Multi-Stage decision Making تبدیل شد

- We may not know where the goal is.

Dynamic Optimization: • Multi-Stage Decision Making

- We have a new variable at each stage which is named STATE.
- A decision at an state can lead to specific set of states.
- The state can also affect the cost function.
- The transition between states in two consecutive stages depend on the previous state(s) and corresponding decision.

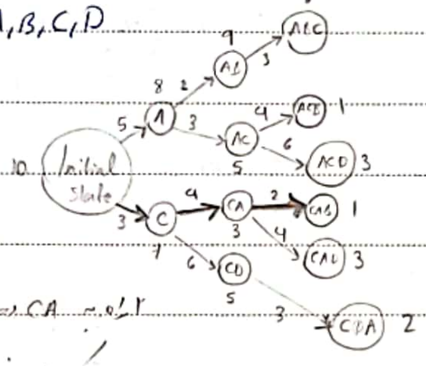
(ادامه در صفحه بعد)

Subject :

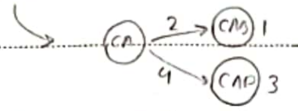
Year . Month . Date .

حال یک مثال از dynamic optimization (→ Programming) را بررسی می‌کنیم؛

Scheduling Example: Find optimal sequence of operations A, B, C, D



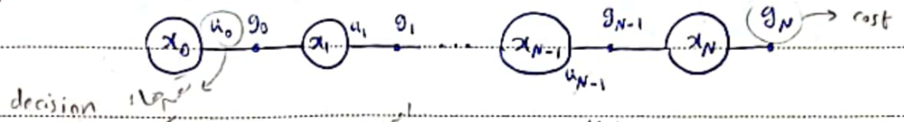
- Start from the last tail subproblem and go backwards
- At each state-time pair, we record the optimal cost-to-go and the optimal decision.



۲ برای CA بهتر است زیرا هزینه ۳ دیگری هزینه ۲ را دارد

جهت اینکه مجدداً cost-to-go را حساب نکنیم!

Formulating Dynamic Optimization problem:



با بردن از حالت و تصمیم، هزینه g را دارد.

$$\underset{u_0, u_1, \dots, u_{N-1}}{\text{Min}} \; E\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k) \right\}$$

اگر w نباشد، deterministic می‌شود.

با بودن در حالت x_k و تصمیم u_k ... با احتمال p به x_k ... با احتمال $1-p$ به ... stochastic می‌شود.

می‌شود این الگوریتم به ...

حال می‌خواهیم به صورت کلی بیان کنیم؛

- **System:** $x_{k+1} = f_k(x_k, u_k, w_k)$, $k = 0, \dots, N-1$
- **Control constraints:** $u_k \in U_k(x_k)$
- **Probability distribution:** $P_k(\cdot \mid x_k, u_k)$ of w_k
- **Policies:** $\pi = \{\mu_0, \dots, \mu_{N-1}\}$, where μ_k maps states x_k into controls $u_k = \mu_k(x_k)$ and is such that $\mu_k(x_k) \in U_k(x_k)$ for all x_k (استراتژی‌های ما)

- **Expected cost of π starting at x_0 is:** $J_\pi(x_0) = E\left\{ g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k), w_k) \right\}$

- **Optimal cost function:** $J^*(x_0) = \min_\pi J_\pi(x_0)$

بنابراین DP Algorithm به صورت زیر می‌شود؛

- Start with: $J_N(x_N) = g_N(x_N)$
- Go backwards using: $J_k(x_k) = \min_{u_k \in U_k(x_k), w_k} E\{ g_k(x_k, u_k, w_k) + J_{k+1}(f_k(x_k, u_k, w_k)) \}$ $k = 0, 1, \dots, N-1$

Subject :

Year . Month . Date .

Reinforcement Learning :

چیزی بین RL و DP

بهینه‌سازی دینامیکی در برنامه‌ریزی پویا، تقریبی، در ادامه آنچه جزو اهداف طرح این بحث بیان می‌کنیم

- سیستم بهینه‌سازی دینامیکی داریم. - حالت نهایی مشخص نیست یا حالت نهایی وجود ندارد

- با حالت ابتدای سیستم، انتهای هدف را می‌دانیم - تعداد مراحل تصمیم‌گیری نامعلوم یا نامحدود باشد

- با حالت ابتدای سیستم اطلاعاتی در مورد تغییر حالت نداریم - تعداد حالت‌های ممکن در هر مرحله تصمیم‌گیری بسیار زیاد باشد با استفاده از DP با مشکل مواجه (curse of dimensionality)

تعریف : فرآیند تصمیم مارکوف (MDP) : خاصیت مارکوف بودن به این معنی است که حالت و پاداش عامل در مرحله t+1 بعد فقط به حالت و کنش عامل در زمان t بستگی دارد.

احتمال اینکه حالت در t+1، s' و جریمه یا پاداش برابر r باشد:

$$Pr\{s_{t+1}=s', r_{t+1}=r \mid s_t, a_t, s_{t-1}, a_{t-1}, \dots, r_1, s_0, a_0\} = Pr\{s_{t+1}=s', r_{t+1}=r \mid s_t, a_t\}$$

اطلاعات مورد نیاز :

$$P^a_{ss'} = Pr\{s_{t+1}=s' \mid s_t=s, a_t=a\} \quad \text{for all } s,s' \in S,\ a \in A(s),$$

مدل دینامیک سیستم

این اطلاعات به صورت closed form نیست یعنی یک تابع f_t نداریم که بتوانیم آن را حل کنیم.

احتمال رفتن agent از حالت s به s' تحت عمل a

$$R^a_{ss'} = E\{r_{t+1} \mid s_t=s, a_t=a, s_{t+1}=s'\} \quad \text{for all } s,s' \in S,\ a \in A(s)$$

مدل تابع هدف

reward ای که از رفتن agent از حالت s به s' تحت عمل a دریافت می‌شود

استراتژی و هدف : ■ در هر مرحله عامل باید تناظری بین حالت‌ها و اعمال برقرار کند. احتمال انتخاب هر عمل ممکن برقرار می‌شود.

این تناظر "استراتژی" یا "سیاست" عامل نام دارد که همان متغیر بهینه‌سازی است.

$$\pi_t(s,a) = prob.(a_t=a \text{ when } s_t=s)$$

■ عامل باید سیاست خود را طوری تعیین کند تا به هدف خود که بیشینه‌سازی مجموع پاداش‌هایی است که در طول زمان به دست می‌آورد برسد

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

Batus

Discount factor → $0 \le \gamma \le 1$: این عامل نشان می‌دهد که reward ها در زمان آینده چقدر اهمیت دارند

Subject :

Year . Month . Date .

مدل سازی تابع هدف: تابع ارزش

- میزان ارزش یک حالت به پاداش هایی که با شروع از آن حالت انتظار می رود گفته می شود. ارزش s تحت استراتژی π (π = policy)

$$V^{\pi}(s) = E_{\pi}\{R_t \mid s_t=s\} = E_{\pi}\Big\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1} \mid s_t=s\Big\} = E_{\pi}\{r_{t+1} + \gamma V^{\pi}(s_{t+1}) \mid s_t=s\} = \sum_a \pi(s,a)\sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V^{\pi}(s')]$$

- میزان ارزش یک عمل در حالت s به پاداش هایی که با شروع از آن حالت با اتخاذ آن عمل انتظار می رود گفته می شود.

تابع state-action با استراتژی π

$$Q^{\pi}(s,a) = E_{\pi}\{R_t \mid s_t=s, a_t=a\} = E_{\pi}\Big\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1} \mid s_t=s, a_t=a\Big\} = \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V^{\pi}(s')]$$

$$= E_{\pi}\{r_{t+1} + \gamma V^{\pi}(s_{t+1}) \mid s_t=s, a_t=a\}$$

در اینجا دیگر $\pi(s,a)$ را نداریم زیرا a را fix در نظر گرفته ایم

روش های محاسباتی بهینه نیاز داریم: چون همه حالت ها را نمی دانیم نمی توانیم از DP استفاده کنیم

حالت سیستم را به طور کامل نمی شناسیم

(★)

اگر ارزش $V^{\pi}(s)$ معلوم باشد، آنگاه استراتژی ای بهینه است که بتواند پاداش های مورد انتظار را بیشینه کند.

سیاست حریصانه (greedy): انتخاب عملی که متوسط پاداش های مورد انتظار را بیشینه کند با احتمال ۱ و بقیه عمل ها با احتمال صفر

$$\pi^*(s,a) = \underset{a}{\mathrm{Argmax}}\ Q^{\pi}(s,a)$$

رابطه بهینگی بلمن (اصل بهینگی بلمن): نشان می دهد استراتژی بهینه چه باشد

$$V^*(s) = \max_a E_{\pi}\{r_{t+1} + \gamma \cdot V^*(s_{t+1}) \mid s_t=s, a_t=a\} = \max_a \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma \cdot V^*(s')]$$

$$= \max_a Q^*(s,a)$$

Batus

Subject :

Year . Month . Date .

(★) به دلیل اینکه $V^{\pi}(s)$ الزاماً ارتباطات معلوم نیست به برنامه‌ریزی پویای تکراری روی آورده‌ایم :

برنامه‌ریزی پویای تکراری : ▪ روش بازگشتی برای سیاست (Policy Iteration) (PI)

این روش از ۲ بخش تشکیل شده است : ۱) ارزش یابی سیاست : به ازای یک سیاست دلخواه ارزش حالت‌ها را مشخص می‌کند

$$V_{k+1}^{\pi}(s) = E_{\pi}\{r_{t+1} + \gamma \cdot V_k^{\pi}(S_{t+1}) \mid s_t = s\} = \sum_a \pi(s,a) \sum_{s'} P_{ss'}^{a} [R_{ss'}^{a} + \gamma \cdot V_k^{\pi}(s')]$$

← به طور عادی چطوری استفاده میشه؟ چون تمام ارزش حالات را نداریم، مجبوریم آن را Initialize کنیم.

← k نشان‌دهنده شماره iteration می‌باشد.

۲) بهبود سیاست : اگر مرحله اول را آنقدر انجام دادیم که دیگر ارزش حالات تغییری نکرد، به سیاستی به انتخاب بهترین عمل بهبود ببخشیم.

$$\pi_{k+1}(s) = \underset{a}{\arg\max}\, Q^{\pi_k}(s,a) = \underset{a}{\arg\max} \sum_{s'} P_{ss'}^{a} [R_{ss'}^{a} + \gamma V(s')]$$

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization : $V(s) \in R$ and $\pi(s) \in A(s)$ arbitrarily for all $s \in S$; $V(terminal) \doteq 0$

2. Policy Evaluation : Loop:

  $\Delta \leftarrow 0$

  Loop for each $s \in S$:

    $v \leftarrow V(s)$

    $V(s) \leftarrow \sum_a \pi(s,a) \sum_{s'} P_{ss'}^{a} [R_{ss'}^{a} + \gamma \cdot V(s')]$

    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$

  until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement : policy_stable ← True

  For each $s \in S$:

    old_action ← $\pi(s)$

    $\pi(s) \leftarrow \underset{a}{\arg\max} \sum_{s'} P_{ss'}^{a} [R_{ss'}^{a} + \gamma V(s')]$

    If old_action $\neq \pi(s)$, then policy_stable ← false

  If policy_stable , then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Subject :

Year . Month . Date .

Value Iteration

روش بازگشتی بر اساس ارزش (VI):

1) $V_{k+1}(s) = \underset{a}{Max}\, E_{\pi}\{ r_{t+1} + \gamma . V_k^{\pi}(s_{t+1}) \mid s_t = s, a_t = a \}$

$$= \underset{a}{Max} \sum_{s'} P_{ss'}^{a} [R_{ss'}^{a} + \gamma . V_k^{\pi}(s')] = \underset{a}{Max}\, Q_{k+1}(s,a)$$

2) $\pi^*(s) = \underset{a}{\arg\max}\, Q^*(s,a)$

Value Iteration, for estimating $\pi \approx \pi_*$:

Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation Initialize $V(s)$, for all $s \in S^+$, arbitrarily except that $V(terminal) = 0$:

Loop:

$\Delta \leftarrow 0$

Loop for each $s \in S$:

$v \leftarrow V(s)$

$V(s) \leftarrow \underset{a}{max} \sum_{s'} P_{ss'}^{a} [R_{ss'}^{a} + \gamma V(s')]$

$\Delta \leftarrow max(\Delta, | v - V(s) |)$

until $\Delta < \theta$

Output a deterministic policy, $\pi \approx \pi_*$, such that $\pi(s) = \underset{a}{\arg max} \sum_{s'} P_{ss'}^{a} [R_{ss'}^{a} + \gamma V(s')]$

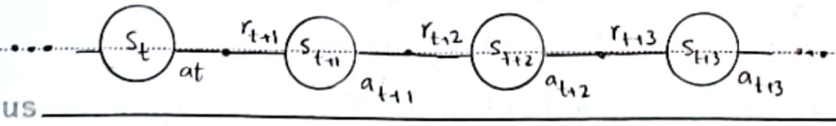
یادگیری تقویتی: بدون مدل یا تفاوت زمانی

→ مدل مسئله در دست نیست ($P_{ss'}^{a}$, $R_{ss'}^{a}$)

→ عامل با سعی و تعامل با محیط و دریافت اطلاعات عددی از حالت و پاداش سعی در بیشینه کردن سود خود خواهد داشت.

روش مونت کارلو:

→ بدانیم روش باید از حالت‌ها و دریافت پاداش‌ها سعی در بهبود تخمین خود از $Q(s,a)$ داریم:



Subject :

Year .　Month .　Date .

- در هر اپیزود، بازگشتی از یک حالت و دنبال کردن یک سیاست مجموع پاداش‌های دریافتی محاسبه می‌شود.
و تصمیم a را گرفته‌ایم

$Return_i = r_1 + \gamma r_2 + \gamma^2 r_3 + \dots + \gamma^{k-1} r_k \rightarrow$ مثلاً در حالت m قرار داریم و تا پایان task، k بار حرکت دیگر داریم با پاداش
حالت m و تصمیم a برابر Return_i خواهد بود.

- میانگین پاداش‌های دریافت شده در اپیزودهای مختلف و بازگشتی از جفت هر حالت (s,a) دور هر حرکت به عنوان
تخمینی از Q(s,a) در نظر گرفته می‌شود:　$Q(s,a) \leftarrow \underset{i}{\text{Average}}(Return_i)$

حال می‌خواهیم ببینیم چه سیاستی را به چه شکلی باید انتخاب کنیم.

Exploration-Exploitation Balance:

- سیاست عامل بین کاوش و بهره‌برداری تعادل ایجاد می‌کند.

- نقش سیاست هم به عنوان متغیر بهینه‌سازی و هم به عنوان عامل سازنده محیط.

ε_Greedy　　　　　　　　　　　　　　　　Softmax

بهینه action　　　　　　　　　　　　$Q(a) \rightarrow$ ارزش تصمیم a در حالت s

$$\pi(s,a) = \begin{cases} 1-\varepsilon+\frac{\varepsilon}{|A(s)|} & \text{if } a = a^* \\ \frac{\varepsilon}{|A(s)|} & \text{if } a \neq a^* \end{cases}$$

$$\pi(s,a) = \frac{e^{Q(a)/\tau}}{\sum_{b \in A(s)} e^{Q(b)/\tau}}$$

این سیگما برای این است

در هر حالت بهترین action ای که می‌توانیم استفاده کنیم چه است؟ به آن احتمال $1-\varepsilon+\frac{\varepsilon}{|A(s)|}$ را داده و به بقیه تصمیمات احتمال $\frac{\varepsilon}{|A(s)|}$ را نسبت می‌دهیم.

در ابتدا $\tau \to \infty$ میل می‌دهیم تا احتمال انتخاب همه actionها برابر باشد. با افزایش epochها، $\tau \to 0$ میل پیدا می‌کند، آنگاه هر تصمیم Q(a) بیشتری داشته باشد احتمال بالایی دارد. A(s): مجموعه همه actionها

ε در ابتدا باید برابر (به سمت) ۱ باشد و با افزایش epochها از آن کاسته شود (به سمت صفر).

State ←	a_0	a_1	a_2	a_3 → Action
s_0	Q_{00}	Q_{01}	Q_{02}	Q_{03}
s_1	Q_{10}	Q_{11}	Q_{12}	Q_{13}
s_2	Q_{20}	Q_{21}	Q_{22}	Q_{23}

Subject :

Year . Month . Date .

در هر اپیزود، با شروع از یک حالت و دنبال کردن یک سیاست مجموع پاداش های دریافتی محاسبه می‌شود.

و تصمیم a را می‌گیریم

$Return_i = r_1 + \gamma r_2 + \gamma^2 r_3 + \dots + \gamma^{k-1} r_k$ ← شروع در حالت m کار را شروع کرده تا پایان task یک بار به حداکثر رسیدیم ارزش

حالت m و تصمیم a برابر $Return_i$ خواهد بود.

میانگین پاداش های دریافت شده در اپیزودهای مختلف با شروع از جفت حالت (s,a) در هر حرکت به عنوان

تخمینی از $Q(s,a)$ در نظر گرفته می‌شود: $Q(s,a) \leftarrow \underset{i}{Average}(Return_i)$

حال می‌خواهیم ببینیم چه سیاستی را به عنوان بهترین انتخاب کنیم.

Exploration-Exploitation Balance:

سیاستی که بین کاوش و بهره برداری تعادل ایجاد می‌کند.

نقش سیاست هم به عنوان تعیین بهینه سازی و هم به عنوان عامل مشاهده محیط.

ε_Greedy

action بهینه

$$\pi(s,a) = \begin{cases} 1-\varepsilon+\frac{\varepsilon}{|A(s)|} & \text{if } a = a^* \\ \frac{\varepsilon}{|A(s)|} & \text{if } a \neq a^* \end{cases}$$

در هر حالت بهترین action ای که می‌توانیم استفاده کنیم چیست؟ به آن احتمال $1-\varepsilon+\frac{\varepsilon}{|A(s)|}$ را داده و به بقیه تصمیمات احتمال $\frac{\varepsilon}{|A(s)|}$ را نسبت می‌دهیم.

Softmax

ارزش تصمیم a در حالت s

$$\pi(s,a) = \frac{e^{Q(a)/\tau}}{\sum_{b\in A(s)} e^{Q(b)/\tau}}$$

این بخش برای اینکه

در ابتدا τ را بزرگ می‌دهیم تا احتمال انتخاب همه action ها برابر باشد.

اگر با افزایش epoch ها، τ را کاهش دهیم، آنگاه هر تصمیم Q(a) بیشتری داشته باشد احتمال را بالا می‌برند.

A(s): مجموعه همه action ها

ε در ابتدا باید مقداری نزدیک به یک داشته باشد و با افزایش epoch ها از آن کاسته شود (به سمت صفر).

State \ Action	a_0	a_1	a_2	a_3
s_0	Q_{00}	Q_{01}	Q_{02}	Q_{03}
s_1	Q_{10}	Q_{11}	Q_{12}	Q_{13}
s_2	Q_{20}	Q_{21}	Q_{22}	Q_{23}

Subject :

Year . Month . Date .

A Monte Carlo algorithm

Algorithm parameter : small $\varepsilon > 0$

Initialize :

$\pi \leftarrow$ an arbitrary ε-soft policy

$Q(s,a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Repeat forever (for each episode) :

Generate an episode following π : $S_0, A_0, \overset{\text{reward}}{R_1}, \dots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode , $t = T-1, T-2, \dots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1, \dots, S_{t-1}, A_{t-1}$:

Append G to $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow average(Returns(S_t, A_t))$

$A^* \leftarrow \underset{a}{\arg\max}\, Q(S_t, a)$

For all $a \in \mathcal{A}(S_t)$:

$$\pi(a|S_t) \leftarrow \begin{cases} 1 - \varepsilon + \dfrac{\varepsilon}{|\mathcal{A}(S_t)|} & \text{if } a = A^* \\ \dfrac{\varepsilon}{|\mathcal{A}(S_t)|} & \text{if } a \neq A^* \end{cases}$$

مقایسه : (مونت کارلو و DP)

- این روش با تعامل مستقیم با محیط عمل می‌کند در حالتی که مدلی از محیط وجود ندارد

- ساده‌تر شدن مدل‌سازی و شبیه‌سازی

- امکان توجه بیشتر به مجموعه مشخصی از حالت‌ها در مقابل حالت‌های کم‌ارزش‌تر

- تخمین‌های هر حالت مستقل از حالت‌های دیگر است برخلاف برنامه‌ریزی پویا که تخمین هر حالت بر اساس حالت دیگر می‌باشد

Subject :
Year . Month . Date .

روش‌های تفاضل زمانی (Temporal Difference)

- ایده اصلی روش‌های اختلاف زمانی براساس ترکیب روش برنامه‌ریزی پویا و روش مونت‌کارلو می‌باشد.
- در این روش مانند روش مونت‌کارلو می‌توان بدون داشتن اطلاعات از محیط و با تعامل با آن، عمل یادگیری را انجام داد.
- در این روش براساس ایده برنامه‌ریزی پویا در هر مرحله تخمین خود را براساس تخمین قبلی و مشاهدات جدید بهبود می‌بخشد. (برخلاف) در مقابل روش مونت‌کارلو منتظر رسیدن به حالت نهایی برای بهبود تخمین نمی‌شود که به آن ایده "بوت استرپ" می‌گویند.
- در روش تفاضل زمانی از خطای تخمین مدل عامل در حالت کنونی و مدل عامل پس از انجام عمل و قرار گرفتن در حالت جدید، جهت بهبود تابع ارزش استفاده می‌شود.

$$V(s_t) = E\left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right\} = E\{ r_{t+1} + \gamma V(s_{t+1}) \mid s_t = s \} \rightarrow$$ از dynamic programming الهام گرفته شده

$$\underset{\text{تابع ارزش}}{V_{t+1}(s_t)} = V_t(s_t) + \alpha \cdot \varepsilon_t = V_t(s_t) + \alpha \cdot [\underbrace{r_{t+1} + \gamma V_t(s_{t+1})}_{\hat{V}_t(s_t)} - V_t(s_t)] \qquad 0 \leq \alpha \leq 1$$

در واقع می‌خواهیم $V_{t+1}(s_t)$ را براساس $V_t(s_{t+1})$ و یک خطا تخمین بزنیم. اگر reward زیاد بیشتر باشد که ارزش یک بالا می‌رود!

انواع روش‌ها:
1) روش SARSA
2) روش آموزش Q
3) روش عملگر-نقاد

روش $S\overset{\text{حالت اولیه}}{}A\overset{\text{action}}{}R\overset{\text{reward}}{}S'\underset{\text{حالت ثانویه}}{}A'$ (action): عمل a را براساس سیاست (soft-max , ε-greedy) خود انتخاب می‌کند، پاداش r را دریافت می‌کند و به حالت s' می‌رود و در نهایت دوباره براساس سیاست خود عمل a' را انجام می‌دهد. پس از این مراحل تابع ارزش عمل (Q) به شکل زیر تجدید می‌شود:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \cdot [r + \gamma \cdot Q(s',a') - Q(s,a)]$$

Subject :

Year .   Month .   Date .

Sarsa: An on policy TD control algorithm

Algorithm parameters: step size $\alpha \in (0,1]$, small $\varepsilon > 0$

Initialize $Q(s,a)$, for all $s \in S^+$, $a \in A(s)$, arbitrarily expect that $Q(terminal, \cdot) = 0$ (*)

Loop for each episode:

  Initialize S

  Choose A from S using policy derived from Q (e.g., ε-greedy)

  Loop for each step of episode:

    Take action A, observe R, S'

    Choose A' from S' using policy derived from Q (e.g., ε-greedy)

    $Q(S,A) \leftarrow Q(S,A) + \alpha [R + \gamma Q(S',A') - Q(S,A)]$

    $S \leftarrow S'$; $A \leftarrow A'$;

  until S is terminal

روش Q: در این روش برخلاف روش SARSA مقادیر ارزش عمل براساس عمل حریصانه جهت

بروزرسانی مورد استفاده قرار میگیرد و بروزرسانی پیش از اتخاذ عمل بعدی (a') انجام خواهد شد.

$$Q(s,a) \leftarrow Q(s,a) + \alpha [r + \gamma \cdot \max_{a'} Q(s',a') - Q(s,a)]$$

Q-learning: An off-policy TD algorithm

(*)

Loop for each episode:

  Initialize S

  Loop for each step for episode:

    Choose A from S using policy derived from Q (e.g., ε-greedy)

    Take action A, observe R, S'

    $Q(S,A) \leftarrow Q(S,A) + \alpha [R + \gamma \max_a Q(S',a) - Q(S,A)]$

    $S \leftarrow S'$

until S is terminal

Batus

Subject :

Year . Month . Date .

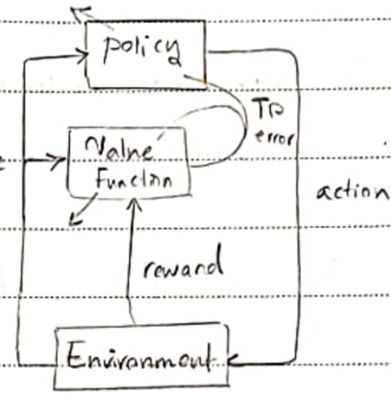
روش عملگر - نقاد Actor_Critic :

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$$

The value of the next state may give more insight about how the last action was good rather than blending this value to the action taken in the next step.

$$\pi_t(s,a) = Pr\{a_t = a \mid s_t = s\} = \frac{e^{p(s,a)}}{\sum_b e^{p(s,b)}} , \quad p(s_t, a_t) \leftarrow p(s_t, a_t) + \beta \cdot \delta_t$$

$$V_{t+1}(s_t) = V_t(s_t) + \alpha \delta_t = V_t(s_t) + \alpha \cdot [r_{t+1} + \gamma V_t(s_{t+1}) - V_t(s_t)]$$



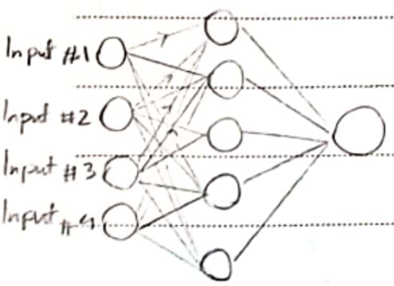
Continues RL:

اگر تعداد حالت‌ها و action های ما خیلی زیاد باشند، روش‌های معرفی شده از لحاظ زمانی بسیار پرهزینه خواهند بود

(اگر محیط را تصادفی بگیریم، هر بار ممکن است به یک حالت جدید برسیم و یک reward متفاوت بگیریم.) function Approximator

کاری که می‌توان انجام داد این است که از جدول "state_action" استفاده نکنیم؛ در واقع یک شبکه عصبی (یک تابع تخمین‌گر را)

تشکیل دهیم که حالت و action را ورودی بگیرد و Q آن را خروجی دهد.



$\rightarrow$ Output $= Q(s,a;w)$ وزن‌های شبکه عصبی

مبنای این روش، سیاست‌ها است. policy ها به سمت سیاست بهینه خواهد برد $\pi(s,a;\theta)$

حال به جای یک سیاست بهینه، مدل parameterize کردن سیاست را در نظر بگیریم.

Subject :

Year . Month . Date .

Advantages of continuous policy parameterization.

- The action probabilities change smoothly as a function of the learned parameter, whereas in epsilon-greedy selection the action probabilities may change dramatically for an arbitrarily small change in the estimated action values.

- This is a good way of injecting prior knowledge about the desired form of the policy into reinforcement learning system.

- The policy can be parameterized in any way, as long as is differentiable with respect to its parameters.

- Offers useful ways of dealing with continous action spaces.

- Although, the temerature or Epsilon could be reduced over time to approach determinism, but in practice it would be difficult to choose the reduction schedule, or even the initial temprature

Policy Gradient: policy Objective Function

- Goal: given policy $\pi_\theta(s,a)$ with parameters θ, find best θ
- But how do we measure the quality of a policy π_θ?

In episodic environments we can use the start value: $J_1(\theta) = V^{\pi_\theta}(s_1) = E_{\pi_\theta}[v_1]$

In continuing environments we can use the average value:

$$J_{avV}(\theta) = \sum_s d^{\pi_\theta}(s) V^{\pi_\theta}(s)$$

Distribution of Markov chain for π_θ

Or the average reward per time-step.

$$J_{avR}(\theta) = \sum_s d^{\pi_\theta}(s) \sum_a \pi_\theta(s,a) R_s^a$$

Subject :

Year .   Month .   Date .

Policy Gradient : Score Function

The score function is $\nabla_\theta \log \pi_\theta(s,a)$

$$\nabla_\theta \pi_\theta(s,a) = \pi_\theta(s,a) \frac{\nabla_\theta \pi_\theta(s,a)}{\pi_\theta(s,a)} = \pi_\theta(s,a) \nabla_\theta \log \{\pi_\theta(s,a)\}$$

We will use a softmax policy as a running example.

Weight actions using linear combination of features $\phi(s,a)^T\theta$

Probability of action is proportional to exponentiated weights:

$$\pi_\theta(s,a) \propto e^{\phi(s,a)^T\theta} = \frac{e^{\phi(s,a)^T\theta}}{\sum_b e^{\phi(s,b)^T\theta}}$$

The score function is : $\nabla_\theta \log \pi_\theta(s,a) = \phi(s,a) - E_{\pi_\theta}[\phi(s,.)]$

Compute the policy gradient:

Consider a simple class of one-step MDPs

- Starting in state $s \sim d(s) \rightarrow$ احتمال دیدن حالت s با ← تابعی از سیاست نمی‌باشد (نرمال می‌شود)
- Terminating after one time-step with reward $r = R_{s,a}$

$$J(\theta) = E_{\pi_\theta}(r) = \sum_{s \in S} d(s) \sum_{a \in A} \pi_\theta(s,a) R_{s,a}$$

$$\nabla_\theta J(\theta) = \sum_{s \in S} d(s) \sum_{a \in A} \pi_\theta(s,a) \nabla_\theta \log \pi_\theta(s,a) R(s,a) = E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) r]$$

Policy Gradient Theorem:

The policy gradient theorem generalizes the approach to multi-step MDPs.

Replaces intantaneous reward r with long-term value $Q^\pi(s,a)$.

Theorem:

For any differentiable policy $\pi_\theta(s,a)$, for any of the policy objective function $J = J_1$, J_{avR}, or $\frac{1}{1-\gamma} J_{avV}$, the policy gradient is:

$$\nabla_\theta J(\theta) = E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) Q^{\pi_\theta}(s,a)]$$

Monte Carlo policy Gradient

→ Update parameters by stochastic gradient ascent

→ Using policy gradient theorem

→ Using return v_t as an unbiased sample of $Q^{\pi_\theta}(s_t, a_t)$

reward

$$\Delta\theta_t = \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$$

function REINFORCE
  Initialize θ arbitrarily
  for each episode $\{s_1, a_1, r_2, \ldots, s_{T-1}, a_{T-1}, r_T\} \sim \pi_\theta$ do
    for $t = 1$ to $T-1$ do
      $\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$
    end for
  end for
  return θ
end function

Monte_Carlo policy gradient still has high variance

We use a critic to estimate the action_value function, $Q_w(s,a) \approx Q^{\pi_\theta}(s,a)$

Actor_critic algorithms maintain two sets of parameters

Critic Updates action_value function parameters w

Actor Updates policy parameters θ, in direction suggested by critic

Actor_critic algorithm follow an approximate policy gradient

$$\nabla_\theta J(\theta) \approx E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)]$$

$$\nabla\theta = \alpha \nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)$$

Action_Value Function approximation

Approximate the action_value function → q بجای Q برابر : $\hat{q}(S,A,w) \approx q_\pi(S,A)$

Minimise mean_squared error between approximate action_value fn $\hat{q}(S,A,w)$ and true action_value fn $q_\pi(S,A)$

$$J(w) = E_\pi[(q_\pi(S,A) - \hat{q}(S,A,w))^2]$$

Subject :

Year . Month . Date .

Monte Carlo policy Gradient

→ Update parameters by stochastic gradient ascent

→ Using policy gradient theorem

→ Using return v_t as an unbiased sample of $Q^{\pi_\theta}(s_t, a_t)$

reward

$$\Delta\theta_t = \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$$

```
function REINFORCE
    Initialize θ arbitrarily
    for each episode {s_1, a_1, r_2, ..., s_{T-1}, a_{T-1}, r_T} ~ π_θ do
        for t=1 to T-1 do
            θ ← θ + α ∇_θ log π_θ(s_t, a_t) v_t
        end for
    end for
    return θ
end function
```

Monte_Carlo policy gradient still has high variance

We use a critic to estimate the action_value function, $Q_w(s,a) \approx Q^{\pi_\theta}(s,a)$

Actor_critic algorithms maintain two sets of parameters

Critic Updates action_value function parameters w

Actor Updates policy parameters θ, in direction suggested by critic

Actor_critic algorithm follow an approximate policy gradient

$$\nabla_\theta J(\theta) \approx E_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)]$$

$$\nabla\theta = \alpha \nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)$$

Action_Value Function approximation.

Approximate the action_value function → q همان Q است : $\hat{q}(S,A,w) \approx q_\pi(S,A)$

Minimise mean_squared error between approximate action_value fn $\hat{q}(S,A,w)$ and true action_value fn $q_\pi(S,A)$

$$J(w) = E_\pi[(q_\pi(S,A) - \hat{q}(S,A,w))^2]$$

Use stochastic gradient descent to find a local minimum:

$$-\frac{1}{2}\nabla_w J(w) = (q_\pi(S,A) - \hat{q}(S,A,w))\nabla_w \hat{q}(S,A,w)$$

$$\Delta w = \alpha(q_\pi(S,A) - \hat{q}(S,A,w))\nabla_w \hat{q}(S,A,w)$$

Like prediction, we must substitute a target for $q_\pi(S,A)$.

→ For MC, the target is the return G_t.

$$\nabla w = \alpha(G_t - \hat{q}(S_t, A_t, w))\nabla_w \hat{q}(S_t, A_t, w)$$

→ For TD (0), the target is the TD target $R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})$

$$\nabla w = \alpha(R_{t+1} + \gamma \hat{q}(S_{t+1}, A_{t+1}, w) - \hat{q}(S_t, A_t, w))\nabla_w \hat{q}(S_t, A_t, w)$$

Actor-Critic policy Gradient

→ Simple actor-critic algorithm based on action-value critic

→ Using linear value fn approx. $Q_w(s,a) = \phi(s,a)^T w$

Critic: Updates w by TD(0)

Actor: Updates θ by policy gradient

function QAC:

Initialise s, θ, w

Sample $a \sim \pi_\theta$

for each step do

Sample reward $r = R_s^a$; Sample transition $s' \sim P_s^a$

Sample action $a' \sim \pi_\theta(s', a')$

$\delta = r + \gamma Q_w(s', a') - Q_w(s,a)$

$\theta = \theta + \alpha \nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)$

$w \leftarrow w + \beta \delta \phi(s,a)$

$a \leftarrow a'$, $s \leftarrow s'$

end for

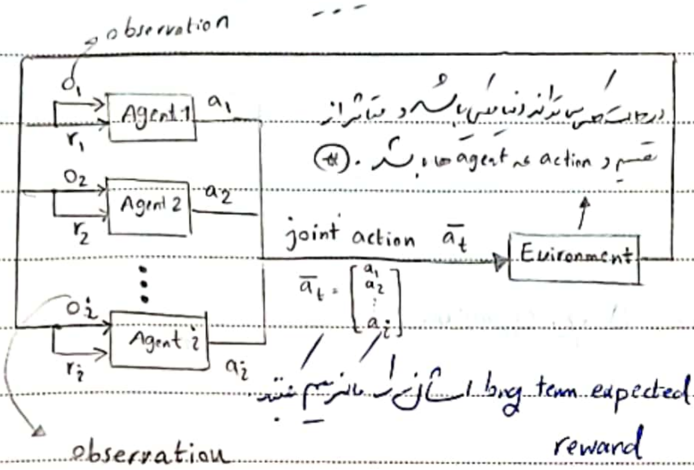
end function

Subject :

Year . Month . Date .

Multi Agent Reinforcement Learning

(*) به طور جدی حالت دنیا (محیط) با $\bar{a}_t$ از تعاملی به عوامل دیگر تغییر می‌کنند (s و s').



1) حاصل joint action هر agent ها

joint state s_t , reward $\bar{r}_t$

2) هر agent فقط reward خود را شامل می‌کند

تعاملی تصمیم‌گیرنده با agent طراحی به هدف هر یک این است که long term expected reward آن را ماکزیمم کنند

(این می‌توانند task Competitive یا Cooperative باشد)

Multi Agent MDP

این سیستم مدل‌سازی می‌شود به action هر agent و reward ای که دریافت می‌کنند و در انتقال حالت دخیل هست

state transition

فارغ از اینکه الگوریتم یادگیری هر agent چی هست!

Set of agents $1, \dots, n$ $\quad (N, S, \vec{A}, \vec{R}, T) \rightarrow T : S \times \vec{A} \rightarrow \Pi(S)$

Set of states

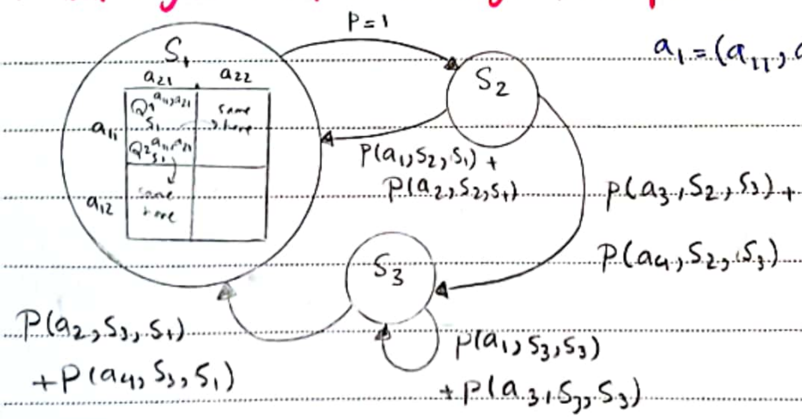
Stochastic transition function, specifying the probability of the next state

$\vec{A} = A_1 \times A_2 \times \dots \times A_n$ $\quad \vec{R} = R_1 \times R_2 \times \dots \times R_n$

A_i : set of actions of agent i $\quad$ تابع سود $R_i : S \times \vec{A} \rightarrow R$

immediate reward function of agent i.

Multi Agent MDP (Markov game) example:



$a_1 = (a_{11}, a_{21})$, $a_2 = (a_{12}, a_{21})$, $a_3 = (a_{11}, a_{22})$, $a_4 = (a_{12}, a_{22})$

$P(a_3, S_2, S_3) + P(a_4, S_2, S_3)$

Subject :

Year . Month . Date .

Learning in Multi-agent MDPs: The simplest way to extend MDPs to the multi-agent

→ MDP is just to add a subscript to Q function of each agent:

$$Q_i(s, a_i) \leftarrow (1-\alpha)\, Q_i(s, a_i) + \alpha\left[R_i(s, \vec{a}) + \gamma V_i(s')\right]$$

e.g. Q learning update: $V_i(s) \leftarrow \max_{a_i \in A_i} Q_i(s, a_i)$

→ That is, to have the learning agent pretend that the environment is passive; ignores the fact that the Q value is a function of the actions selected by the other agents.

→ Assuming each agent can observe the joint action of agents, the first idea is to simply define the Q-values as a function of agents' joint action.

$$Q_i(s, \vec{a}) \leftarrow (1-\alpha)\, Q_i(s, \vec{a}) + \alpha\left[R_i(s, \vec{a}) + \gamma V_i(s')\right]$$

Question: How to evaluate $V(s')$?

→ An agent can use On-policy methods like SARSA to update $V_i(s')$ based on real joint actions in s': $V_i(s') \leftarrow Q_i(s', \vec{a})$

→ However, for off-policy methods we need to have an estimate/expectation from the action of other agents.

→ In any case, we need such estimation/expectation for action selection based on a policy (e.g. greedy)

minimax Q-learning algorithm: For selfish agents, Littman suggests the minimax-Q learning algorithm, in which V is updated with the minimax of Q values:

$$V_1(s) \leftarrow \max_{P_1 \in \Pi(A_1)} \min_{a_2 \in A_2} \sum_{a_1 \in A_1} P_1(a_1) Q(s,(a_1,a_2))$$

در اینجا فرض شده به صورت ۲ تا agent داریم و ارزش حالت را برای agent اول به دست میاوریم. این الگوریتم به

a_2 را بدترین حالت یعنی در نظر میگیریم (یعنی a_2 طوری باشد که به ضرر ما رفتار کند و کمترین reward به ما برسد) سپس P_1 را

طوری پیدا کن که $\sum p_1(a_1) Q(s,(a_1,a_2))$ ماکزیمم شود!

The policy can be updated in ~~the~~ a same way.

Belief based algorithm: One alternative is to explicitly maintain a belief regarding the likelihood of the other agents' policies, and update V based on the induced expectation of the Q-values.

$$V_i(s) \leftarrow \max_{a_i} \sum_{a_{-i} \in A_{-i}} P_i(s, a_{-i})\, Q_i(s,(a_i,a_{-i}))$$

با استفاده از KNN, Histogram (frequency) یا هر pdf estimator ای action های agent دیگر را تخمین زده

سپس P_i را تخمین میزنیم

Friend-or-Foe (FOF) algorithm:

Friend: $V_1(s) \leftarrow \max_{a_1 \in A_1, a_2 \in A_2} Q_1(s,(a_1,a_2))$

Foe: $V_1(s) \leftarrow \max_{P_1 \in \Pi(A_1)} \min_{a_2 \in A_2} \sum_{a_1 \in A_1} P_1(a_1) Q_1(s,(a_1,a_2))$

The idea is simply that i's friends are assumed to work together to maximise i's value

Subject :
Year .   Month .   Date .

while i's foes are working together to minimize i's value.

$$V_i(s) \leftarrow \max_{x \in \Pi(X_1, \dots \times X_k)} \min_{y_1, \dots y_l \in Y_1 \times \dots \times Y_l} \sum_{x_1, \dots, x_k \in X_1 \times \dots \times X_k} \pi(x_1) \dots \pi(x_k) Q_i[s, x_1, \dots, x_k, y_1, \dots, y_l]$$

Fully Coprative Tasks: ← همه عامل‌ها یکسان reward دارند

The agents have the same reward function and the learning goal is maximize the common discount return.

$$\underbrace{Q_{k+1}}_{\text{بروز}}(\overset{\text{state}}{x_k}, \overset{\text{joint action}}{u_k}) = Q_k(x_k, u_k) + \alpha[\underbrace{r_{k+1}}_{\text{reward}} + \gamma \max_{u'} Q_k(x_{k+1}, u') - Q_k(x_k, u_k)]$$

خطا

u', joint action

برای reward function باید جدول Q در نظر بگیریم؛ اگر تعداد state ها و action ها زیاد باشند آنگاه تعداد Q خیلی زیاد می‌شود
(Curse of dimensionality)

Greedy policy: $\bar{h}_i^*(x) = \arg\max_{u_i} \max_{u_1, \dots, u_{i-1}, u_{i+1}, \dots, u_n} Q^*(x, u)$

→ It breaks ties randomly; different agents may break ties in different ways; Resulting joint action may be suboptimal.

		Agent 2	
Agent 1		2, 2	3, 3
		3, 3	2, 2

The Distributed Q_learning algorithm: works in deterministic problems:
reward function برای هر عامل جداگانه باشد

$$Q_{i,k+1}(x_k, u_{i,k}) = \max\{Q_{i,k}(x_k, u_{i,k}), r_{k+1} + \gamma \max_{u_i} Q_{i,k}(x_{k+1}, u_i)\}$$

که بزرگ‌تر باشد!

ایده این است که هر agent وجود عامل‌های دیگر را ignore می‌کند و صرفاً local Q-table خودش را تغییر می‌دهد با در نظر گرفتن حالت

محیط (x_k) و تصمیم (action) عامل i در مرحله k ($u_{i,k}$)

The local Q_values are updated only when the update leads to an increase in the Q_value.

Batus

Subject :

Year . Month . Date .

The local policy is updated only if the update leads to an improvement in the Q_values

$$\bar{h}_{i,k+1}(x_k) = \begin{cases} u_{i,k} & \text{if } \max_{u_i} Q_{i,k+1}(x_k,u_i) > \max_{u_i} Q_{i,k}(x_k,u_i) \\ \bar{h}_{i,k}(x_k) & \text{otherwise} \end{cases}$$

این همان greedy policy است

یعنی چرا به greedy policy بر نمی‌گردد؟

Under the conditions that the reward function is $\boxed{\text{positive}}$ and $Q_{i,0} = 0 \ \forall i$, the local policies of the agents provably converge to an optimal joint policy w.r.t global Q.

u_k ای را پیدا کنید (از) که optimal joint action باشد برای Q-function همه‌ی عامل‌ها باشد. (در نهایت)

r_{k+1} از آن اکشنی به دست آمده که اثر action های بقیه عامل‌ها هم در آن دخالت می‌دهد

Indirect Coordination Methods:

Bias action selection toward actions that are likely to result in good rewards or returns.

action selection عامل‌ها بایستی به سمتی باشد که برای جمع reward بیشتری به همراه داشته باشد

This steers the agents toward coordinated action selections.

مفهوم Bias این است که بدانیم بقیه عامل‌ها چه models ... coordinated action selection برای رسیدن به reward بیشتر انجام می‌دهند

The likelihood of good values is evaluated using, e.g., method of the other agents' estimated

action های عامل‌های دیگر در طول گذشته

by the learner, or ~~estim~~ statistics of the values observed in the past.

(I) Joint Action Learns (JAL) (static tasks)

یعنی state های مختلفی نداریم ($Q_i(s)$) تأثیری ندارد

$$(I) \quad Q_i(u_i, u_{-i}) = (1-\gamma) Q_i(u_i, u_{-i}) + \gamma r(u_i, u_{-i})$$

Action of agent i

Employ empirical models of the other agents' strategies: → belief based نیز هست

Subject :

Year . Month . Date .

$\hat{\sigma}^i_j(u_j) = \dfrac{c^i_j(u_j)}{\sum_{\tilde{u}_j \in U_j} c^i_j(\tilde{u}_j)}$ تخمین agent i از action های agent j (تصمیم)

$c^i_j(u_j)$ → # times agent i observed agent j taking action u_j

$\sum_{\tilde{u}_j \in U_j}$ ← تعداد کل action هایی که agent j انجام داده است

$$\tilde{Q}_i(u_i) = \sum_{u_{-i}} Q_i(u_i, u_{-i}) \prod_{j \neq i} \hat{\sigma}^i_j(u_j)$$ متوسط‌گیری روی action های سایر

یعنی برای همه ترکیب‌های ممکن از u_{-i} ها، u_i ثابت است! u_{-i} ها را در نظر بگیرند

$$\tilde{Q}_i(u_i) = (1-\gamma)\tilde{Q}_i(u_i) + \gamma r(u_i) \quad \text{(II)}$$

(II) همین (I) است که $Q_i(u_i,u_{-i})$ آن تخمین زده شده است

(II) Frequency Maximum Q-value (FMQ) (static tasks)

Q_i را به صورت تک عامله در نظر بگیرد از ابتدا $Q_i(u_i) = (1-\gamma) Q_i(u_i) + \gamma r(u_i, u_{-i})$ داشت و عامل‌های دیگر را لحاظ نکند

$$\tilde{Q}_i(u_i) = Q_i(u_i) + v \frac{C^i_{max}(u_i)}{C^i(u_i)} r_{max}(u_i)$$

→ $r_{max}(u_i)$ is the maximum reward observed after taking action u_i, $C^i_{max}(u_i)$ counts how many times this reward has been observed.

→ works for deterministic tasks, where variance in the rewards resulting from the agent's actions can only be the result of the other agent's actions.