دانشکده علوم ریاضی
دانشگاه صنعتی شریف

الگوریتم‌های خلاصه‌سازی برای مه‌داده

محمدهادی فروغمنداعرابی
پاییز ۱۳۹۹

احساس فشردگی

جلسه هفدهم

نگارنده: ناهید صارم

۱ مروری بر مباحث گذشته

داشتیم سعی می‌کردیم، احساس فشردگی را حل کنیم. الگوریتم کمینه کردن نرم یک را توضیح دادیم.صورت مسئله چی بود؟ایده‌امون این بود که یک x داشتیم، می‌خواستیم ذخیره‌اش کنیم، به جای x، Πx را ذخیره می‌کنیم، که در آن Π تعداد سطرهایش خیلی کمتر از x است. ولی می‌دانیم x، kتنک است. x، kتنک است به جای x ، Πx را ذخیره می‌کنیم و می‌خواهیم بتوانیم بازسازیش کنیم.چگونه بازسازیش می‌کنیم؟

$$minimize \quad ||z||_۱$$

$$s.t. \quad \Pi z = y$$

می‌خواستیم نشان دهیم این الگوریتم کار می‌کند.کی کار می‌کند؟اگر Πای که استفاده می‌کنیم یک خصوصیات خوبی داشته باشد این الگوریتم کار می‌کند.حال برای اینکه این خصوصیات را بیان کنیم دوتا تعریف ارائه می‌دهیم:

تعریف ۱. *ماتریس $\Pi \in R^{m\times n}$ را $(\varepsilon, k) - restricted \ isometry \ property$ یا RIP می‌گویند، هرگاه برای همه‌ی بردارهای kتنک با نرم اقلیدسی یک داشته باشیم:*

$$۱ - \delta \leq ||\Pi x||_۲^۲ \leq ۱ + \delta \quad (۱)$$

تعریف زیر معادل، نامعادله‌ی (۱) می‌باشد:

$$sup_{T\subset[n]||T|=k} ||I_k - (\Pi^{(T)})^* \Pi^{(T)}|| < \delta$$

مثال ۲. *ماتریس* $\sqrt{\frac{d}{m}}SH$ *که در آن* H *یک* $\frac{1}{\sqrt{d}}\times$ *ماتریس هادامار و* S *یک ماتریس نمونه‌گیری با* $m = \Omega(\delta^{-2}klog^{4}n)$*سطر که هر درایه‌اش* ±1*است،به احتمال مثبت یک ماتریس* RIP *می‌باشد.*

تعریف ۳. *ماتریس* $m \times n$ *،* A *دارای خاصیت* $null-space\ \ property$ *از مرتبه‌ی* k *با ثابت* C *می‌باشد، اگر برای هر* $\eta \in R^{n}$ *که* $A\eta = 0$ *و هر مجموعه‌ی* $T \subset \{1, ..., n\}$ *از اندازه‌ی* k *داشته باشیم:*

$$||\eta||_1 \leq C||\eta_{-T}||_1$$

جایی که $T-$ *مکمل مجموعه‌ی* T *می‌باشد.*

لم ۴. *فرض کنید* A *یک ماتریس* RIP *با مرتبه‌ی* $k(2+c)$ *با ثابت* $\delta, c > 1$ *باشد. آنگاه* A *دارای خاصیت* $nullspace\ \ property$ *از مرتبه‌ی* $2k$ *با خاصیت* $(1-\delta)(1+\delta)\sqrt{2/c} + 1 = C$*می‌باشد.*

اثبات. برای اثبات، ابتدا فرضیات را تعریف می‌کنیم.به ازای هر x که دارای خاصیت $(2+c)k$تنک باشد، ماتریس Π را روی آن اعمال کنیم، خواهیم داشت:

$$(1-\delta) \leq ||\Pi x||_2^2 \leq 1+\delta$$

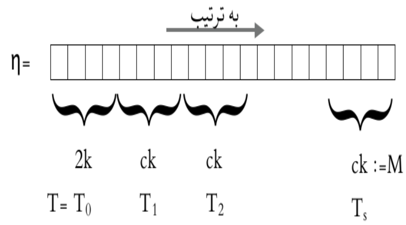
سپس این خاصیت را نیز دارد که اگر η در $null\ space$، Π باشد، آنگاه برای هر $|T| = 2k$ داریم:

$$||\eta||_1 \leq C||\eta_{-T}||_1 \qquad (2)$$

بیان دیگری از نامعادله‌ی (۲) به صورت زیر می‌باشد:

$$||\eta||_1 \leq (C-1)||\eta_{-T}||_1 \qquad (3)$$

حال کافی است (۳) را ثابت کنیم.



می‌خواهیم بگوییم (۳) به ازای هر T درست است.بدترین T، Tای است که سمت راست (۳) را کمینه کند و سمت چپش را بیشینه کند.در نتیجه آن T را در نظر بگیرید که $2k$ تا بزرگترین مولفه‌های η را دارد.این η را می‌توانیم مولفه‌هایش را $sort$ کنید و $2k$تا مولفه‌ی اول را T_0 در نظر بگیرید. از آن به بعد هر ck، ckتا مولفه را $T_1, ..., T_s$ را در نظر بگیرید. اخرین ck را هم M بنامید.حال می‌خواهیم (۳) را ثابت کنیم.اول باید (۳) را به نرم دو تبدیل کنیم، زیرا فرضمان این بود که ماتریس Π نرم دو را ثابت نگه می‌دارد. نرم یک را می‌توان ضرب یک بردار، در بردار یک در نظر بگیریم و سپس از کوشی شوارتز استفاده کنیم.

$$||x||_1 = < x, 1 > \leq ||x||_2\sqrt{2k}$$

در نتیجه داریم:

$$||\eta_T||_1 \leq (2k)^{1/2}||\eta_T||_2$$

حال باید برای نرم دو یک کران پیدا کنیم.برای اینکار T را به دو قسمت $T = T_0 \cup T_1$ تقسیم می‌کنیم.سپس A را وارد می‌کنیم.دقت کنید که A ماتریسی بود که زیاد اندازه بردار را تغیر نمی‌داد، پس داریم:

$$||\eta_T||_2 \leq ||\eta_{T_0} + \eta_{T_1}||_2 \leq \frac{1}{1-\delta}||A(\eta_{T_0} + \eta_{T_1})||_2)$$

حال چون$A\eta = 0$ ، پس داریم:

$$A\eta = 0 \rightarrow A(\eta_{T_0} + \eta_{t_1} + ... + \eta_{t_s}) = 0 \rightarrow A\eta_{T_0} + A\eta_{T_1} = -A\eta_{T_2} - ... - A\eta_{T_s} = \frac{1}{1-\delta}||A(\eta_{T_2} + ... + \eta_{T_s})||_2$$

$$\leq \frac{1}{1-\delta}\Sigma_{j=2}^{s}||A\eta_{T_j}||_2$$

چون A اندازه‌ی kتنک‌ها را زیاد تغیر نمی‌دهد، خواهیم داشت:

$$\leq \frac{1}{1-\delta}\Sigma_{j=2}^{s}||A\eta_{T_j}||_2 \leq (1-\delta)^{-1}(1+\delta)\Sigma_{j=2}^{s}||\eta_{T_j}||_2$$

در نتیجه داریم:

$$||\eta_T||_2 \leq (1-\delta)^{-1}(1+\delta)\Sigma_{j=2}^{s}||\eta_{T_j}||_2$$

اینجا نرم دو داریم و می‌خواهیم به نرم یک تبدیلش کنیم.به ازای هر η_{T_j} داریم:

$$\forall i \in T_{j+1} \qquad |\eta_i| \leq ||\eta_{T_j}||_1/M$$

پس می‌توان نتیجه گرفت:

$$||\eta_{T_{j+1}}||_2 \leq (M(||\eta_{T_j}||_1/M)^2)^{1/2} = ||\eta_{T_j}||_1/M^{1/2}$$

بنابراین:

$$||\eta_T||_2 \leq (1-\delta)^{-1}(1+\delta)/M^{1/2}\Sigma_{j=1}^{s}||\eta_{T_j}||_1 = (1-\delta)^{-1}(1+\delta)/M^{1/2}||\eta_{-T}||_1$$

چون $||\eta_T||_1 \leq (2k)^{1/2}||\eta_T||_2$ و $||\eta_T||_1 \leq (1-\delta)^{-1}(1+\delta)/M^{1/2}||\eta_{-T}||_1$ پس داریم:

$$||\eta_T||_1 \leq (2k)^{1/2}(1-\delta)^{-1}(1+\delta)/M^{1/2}||\eta_{-T}||_1$$

در نتیجه داریم:

$$||\eta_T||_2 \leq O(\frac{1}{\sqrt{k}})||\eta_{-T}||_1$$

چون $M := ck$ پس داریم:

$$||\eta||_1 \leq [1 + (1-\delta)^{-1}(1+\delta)(2/c)^{1/2}||\eta_{-T}||_1]$$

□

لم ۵. *فرض کنید A یک ماتریس با خاصیت $nullspace\ \ property$از مرتبه $2k$ با ثابت $C < 2$ باشد. آنگاه برای هر x^* که $||x^*||_1$ را کمینه می‌کند مشروط بر اینکه $Ax^* = Ax$ ، خواهیم داشت:*

$$||x - x^*||_1 \leq \frac{2C}{2-C}Err_1^k(x)$$

که در آن $Err_1^k := ||x_{tail(k)}||_1$ می‌باشد.

$x_{tail(k)}$ در قضیه‌ی بالا یعنی اینکه در بردار x ، kبزرگترینها را بریزیم دور هر چیزی که می‌ماند، مقدار نرم یکش چقدر است.
لم بالا چه می‌گوید؟ در لم بالا فرض کنید $\eta = x^* - x$. چون A دارای خاصیت $nullspace\ \ property$ از مرتبه‌ی $2k$است پس به ازای هر مجموعه‌ی T که $|T| = 2k$ داریم:

$$||\eta||_1 \leq C||\eta_{-T}||_1$$

حال لم بالا می‌گوید در این شرایط الگوریتم ما یک x^* بر می‌گرداند که نرم یکش کمینه باشد، $Ax^* = Ax$ و

$$||x - x^*||_1 \leq \frac{2C}{2-C}Err_1^k(x)$$

.

اثبات. یک اثبات مختصر از این لم ارائه می‌دهیم. x^* در میان تمام جوابهای دستگاه $Ax = y$ کمترین نرم یک را دارد، بنابراین به ازای هر x که در $Ax = y$ صدق می‌کند خواهیم داشت:

$$||x^*_T||_1 + ||x^*_{-T}||_1 \leq ||x_T||_1 + ||x_{-T}||_1.$$

چون $\eta = x^* - x$ ، طبق نامساوی مثلثی داریم:

$$||x^*_T||_1 \leq ||x_T||_1 - ||\eta_T||_1$$

$$||x^*_{-T}||_1 \leq -||x_{-T}||_1 + ||\eta_{-T}||_1$$

با استفاده از موارد بالا خواهیم داشت:

$$||x_T||_1 - ||\eta_T||_1 - ||x_{-T}||_1 + ||\eta_{-T}||_1 \leq ||x_T||_1 + ||x_{-T}||_1$$

در نتیجه داریم:

$$||\eta_{-T}||_1 \leq ||\eta_T||_1 + 2||x_{-T}||_1 = ||\eta_T||_1 + 2Err^k_1(x)$$

بنابراین:

$$||\eta_{-T}||_1 - ||\eta_T||_1 \leq 2Err^k_1(x)$$

حال چون $||\eta_T||_1 \leq (C-1)||\eta_{-T}||_1$ پس داریم:

$$||\eta_{-T}||_1 \leq \frac{2}{2-C}Err^k_1(x)$$

□

یک نکته را نیز در حاشیه می‌توان گفت.تا به حال ما یک xداشتیم، ذخیره‌اش کرده بودیم، سپس در مرحله‌ی بازیابی نیز یک x^* پیدا کردیم و یک کران برای فاصله‌ی نرم یک xو x^* دادیم.درباره‌ی فاصله‌ی نرم دو می‌توان به صورت زیر کرانش کرد:

$$||\hat{x} - x||_2 \leq O(\frac{1}{\sqrt{k}})||x_{tail(k)}||_1$$

۲ احساس فشردگی (روش تکراری)

تا اینجا همه چیز خوب بود، فقط تنها چیزی که ناراحتمان می‌کرد این است که این الگوریتمی که برای بازیابی x^* استفاده می‌کنیم یک برنامه‌ریزی خطی است. خوشحال می‌شویم اگر بتوانیم این کار را سریعتر انجام دهیم.برای این کار از نزول گرادیان استفاده می‌کنیم. نزول گرادیان چه می‌گوید؟ می‌گوید یک جواب اولیه بگیرید. به عنوان مثال $z = 0$ را به عنوان جواب اولیه در نظر بگیرید.در $\Pi z = y$ صدق نمی‌کند. کمی تغیرش دهید. بعد مثلا می‌بینید تابع هدف مناسب نمی‌باشد و باید در خلاف جهت گرادیان حرکت کنید.امکان دارد باز جوابی که به دست می‌آید خوب نباشد، مثلا k تنک نباشد. kتنکش کنید و دوباره همین کارها را انجام دهید. اگر آن چیزی که می‌خواهید حل کنید خوب باشد، مثلا ماتریس Π، RIP باشد، آن وقت با سرعت نمایی به جواب نزدیک خواهید شد. اینجا هم همین کار را می‌کنیم. اولا الگوریتمان را عوض می‌کنیم و اجازه می‌دهیم یک ذره خطا هم داشته باشد. یعنی فرض می‌کنیم آن y که ما گرفته‌ایم، اینجوری هست که ما به جای x آمده‌ایم Πx را ذخیره کرده‌ایم و آن زمان که آمده‌ایم ذخیره‌اش کنیم Πx ما با کمی نویز ذخیره شده‌است. پس برای بازیابی باید از مسئله‌ی برنامه‌ریزی خطی زیر استفاده کنیم:

$$minimize \quad ||z||_1$$

$$s.t. \quad ||\Pi z - y||_2 \leq \alpha$$

در این مسئله y به اندازه‌ی α خراب شده است به همین خاطر مسئله‌ی برنامه‌ریزی خطی که برای بازیابی استفاده می‌کنیم به صورت بالا درآمده است.

حالا ما می‌خواستیم یک نکته‌ی بیشتری بگوییم و آن این بود که، یک کاری بکنیم که الگوریتممان سریعتر هم بشود. الگوریتممان تکراری است، ابتدا یک x اولیه می‌گیرد اسمش را می‌گذاریم x^0. از روی x^0، x^1 را می‌سازند سپس از x^1، x^2 را می‌سازند، الی آخر. قبل از اینکه الگوریتم را بیان کنیم، می‌خواهیم یک کرانی برای خوب بودن x^tها میخواهم بدهم.خوب بودن x^tها یعنی چی؟ یعنی x^t فاصله‌اش از جواب اصلی که x هست، چقدر هست؟با استفاده از قضیه‌ی زیر یک کران برای این موضوع می‌دهیم.

قضیه ۶. اگر Π دارای خاصیت $RIP-(\varepsilon, 3k)$ برای $\varepsilon < \frac{1}{4\sqrt{2}}$ باشد، آنگاه $\forall T \geq 1$

$$||x^{[T+1]} - x||_2 \lesssim 2^{-T}||x||_2 + ||x_{tail(k)}||_2 + \frac{1}{\sqrt{k}}||x_{tail(k)}||_1 + ||e||_2$$

در قضیه‌ی بالا $||x^{[T+1]} - x||_2$ فاصله تا جواب است. $\frac{1}{\sqrt{k}}||x_{tail(k)}||_1$ کران قدیمی است همان که در عبارت زیر داشتیم:

$$||\hat{x} - x||_2 \leq O(\frac{1}{\sqrt{k}})||x_{tail(k)}||_1$$

این قضیه می‌گوید خطای ما در الگوریتم تکراری به اندازه‌ی خطای قدیمی به اضافه‌ی یک چیزهایی می‌باشد. $||x_{tail(k)}||_2$ (تقریبا) کران قدیمی است.$||e||_2$ خطای اندازه‌گیری است، مانند همان α است که در مسئله‌ی قبلی داشتیم.در واقع داریم:

$$y = \Pi x + e \qquad ||e||_2 \leq \alpha.$$

$2^{-T}||x||_2$ هزینه‌ای است که در ازای سرعتمان می‌پردازیم، عبارتی که در هر تکرار کم می‌شود همین عبارت است بقیه که همواره ثابت می‌مانند. این خطا به صورت نمایی کم می‌شود به همین خاطر خطای کم اهمیتی است. برای اینکه نشان دهیم که $||x_{tail(k)}||_2$ با $||x_{tail(k)}||_1$ تفاوت چندانی ندارد از لم زیر استفاده می‌کنیم.

لم ۷.

$$||x_{tail(2k)}||_2 \leq \frac{1}{\sqrt{k}}||x_{tail(k)}||_1$$

حال می‌خواهیم نشان دهیم که فاصله‌ی ما تا جواب واقعی به صورت نمایی کم می‌شود. در ابتدا فرض کنید x ، kــتنک است، ولی در حالت کلی که x، kــتنک نیست.ولی این خطا که به خاطر kــتنک نبودن لحاظ می‌شود را در e در نظر می‌گیریم و می‌گوییم e خطای اندازه‌گیری است.برای اینکار مراحل زیر را انجام می‌دهیم:

$$\Pi x + e = \Pi(x_{head(k)} + x_{tail(k)}) + e = \Pi x_{head(k)} + \underbrace{(\Pi x_{tail(k)} + e)}_{\hat{e}}$$

می‌توان نشان داد که $\hat{e}$ نرمش زیاد نمی‌باشد واز این استفاده می‌کند که Π، RIP است. الگوریتم تکراری که می‌خواستیم معرفی کنیم به صورت زیر است: x^1 را صفر در نظر بگیرید. سپس x^{t+1}،از روی x^t با استفاده از مرحله‌ی ۴ ساخته می‌شود.

Algorithm 1 Iterative Hard Thresholding (IHT).

1: **function** IHT($\Pi, y(=\Pi x + e), k, T$)
2: $\quad x^{[1]} \leftarrow 0$
3: $\quad$ **for** $t = 1 \cdots T$ **do**
4: $\quad\quad x^{t+1} \leftarrow H_k(x^{[t]} + \Pi^\top(y - \Pi x^{[t]}))$ ▷ Hard thresholding operator (project $a^{[t+1]}$ on $x_{\text{head}}(k)$)
5: $\quad$ **end for**
6: $\quad$ **return** x^{T+1}
7: **end function**

Hk: k_تنک‌ساز

حال میخواهیم ثابت کنیم که این الگوریتم همگرا است. متغیر r^t را که فاصله تا جواب را نشان می‌دهد، به صورت زیر تعریف می‌شود:

$$r^{[t]} := x - x^{[t]}$$

باید نشان دهم که $r^{[t+1]}$ کم می‌شود.

$$||\gamma^{[t+1]}||_2 = ||x - x^{t+1}||_2 = ... \leq \frac{1}{2}||\gamma^{[t]}||_2 + 3||e||_2 \leq 2^{-t}\underbrace{||x||_2}_{||r^{[0]}||_2} + 3(\underbrace{\Sigma_{i=0}^{t-1}2^{-i}}_{<6})||e||_2 \leq 2^{-t}||x||_2 + 6||e||_2 + \frac{1+\varepsilon}{\sqrt{k}}||x_{tail(k)}||_1$$

۳ احساس فشردگی مدلی

ما تا الان چیکار می‌کردیم به جای x، Πx را ذخیره می‌کردیم و می‌گفتیم اگر بدانیم x، kــ تنک است. فرض کنید $\Omega_{n,k} = \binom{[n]}{k}$ تمام زیرمجموعه‌های kعضوی باشد، می‌گوییم اگر یک نفر به ما kتا عدد بدهد و بگوید کدام یکی از این مجموعه‌های kعضوی مدنظرش هست، آنگاه می‌توانیم دقیقا xرا مشخص کنیم. برای اینکه الگوریتم HIT کار کند باید ماتریس Π، ۳kــ تنکها را حفظ کند. نکته‌ی جالب این هست که به جای kــ تنک بودن می‌توانیم هر شرط دیگری نیز بگذاریم.به جای اینکه بگوییم عضو مجموعه‌های kــ تنک است بگوییم $x \in M$. ماتریس Π، باید به ازای هر سه بردار از M فاصله را حفظ کند و الگوریتم به صورت زیر در می‌آید:

Algorithm 2 Model Based Iterative Hard Thresholding (MB-IHT).

function $\mathrm{IHT}(\Pi, y(= \Pi x + e), k, T)$
  $x^{[1]} \leftarrow 0$
  for $t = 1 \cdots T$ **do**
    $x^{[t+1]} \leftarrow P_{\mathcal{M}}(x^{[t]} + \Pi^\top(y - \Pi x^{[t]}))$ ▷ Hard thresholding operator (project $a^{[t+1]}$ on $\mathcal{M}$)
  end for
  return x^{T+1}
end function