www.MatlabKar.com
متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب

شبکه های عصبی

استاد: محمد باقر منهاج

موضوعات

- **فصل هفتم**
  - مقدمه √
  - مبانی بهینه سازی و نقاط بهینه √
  - روشهای می نیمم سازی √
  - یادگیری ویدرو-هوف و شبکه آدالاین
  - الگوریتم LMS
  - کاربرد شبکه آدالاین در فیترهای تطبیقی

مثالی از الگوریتم SD

- تابع $F(\underline{x}) = x_1^{\ 2} + 9x_2^{\ 2}$ با نقطه حداقل [0 0]

$$\underline{x} = [x_1 \quad x_2]^T$$

$$\underline{x}_0 = [0.5 \quad 0.5]^T$$

$$\nabla F(\underline{x}_0) = \begin{bmatrix} 2x_1 \\ 18x_2 \end{bmatrix}_{\underline{x}=\underline{x}_0} = [1 \quad 9]^T$$

$$\underline{x}_1 = \underline{x}_0 - \alpha g_0 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - 0.05 \begin{bmatrix} 1 \\ 9 \end{bmatrix} = \begin{bmatrix} 0.45 \\ 0.05 \end{bmatrix}$$

$$\underline{x}_2 = \underline{x}_1 - \alpha g_1 = \begin{bmatrix} 0.45 \\ 0.05 \end{bmatrix} - 0.05 \begin{bmatrix} 0.9 \\ 0.9 \end{bmatrix} = \begin{bmatrix} 0.405 \\ 0.005 \end{bmatrix}$$

نکات

- با تغییر نرخ یادگیری شکل مسیر حرکت تغییر می کند.
- اگر نرخ یادگیری زیاد شود سرعت نزدیک شدن به نقطه بهینه زیادتر شده ولی میزان نوسانات بیشتر خواهد شد.
- اگر میزان نرخ یادگیری از حدی بیشتر شود، الگوریتم واگرا می گردد. به این حد نرخ یادگیری پایدار گویند.
- برای توابع درجه دوم ثابت می شود که نرخ یادگیری پایدار برابر است با

$$\alpha < \frac{2}{\lambda_{\max}(A)}$$

A ماتریس هسیان تابع F

- محاسبه نرخ یادگیری پایدار در مثال قبل:

$$A = \begin{bmatrix} 2 & 0 \\ 0 & 18 \end{bmatrix}$$

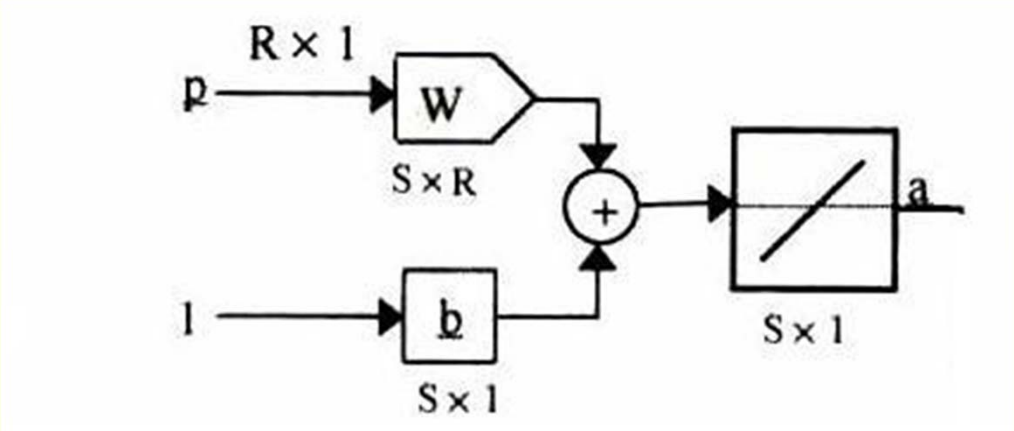
$$\alpha < \frac{2}{\lambda_{\max}(A)} = \frac{2}{18} = 0.11$$

www.MatlabKar.com
متلب کار - مرجع پروژه های آماده و شبیه سازی با متلب

یادگیری LMS یا ویدرو-هوف

شبکه آدالاین

- شبکه آدالاین با قانون یادگیری ویدرو – هوف (معروف به قانون LMS) در سال ۱۹۶۰ و بعد از شبکه پرسپترون با قانون یادگیری SLPR به وجود آمد.
- شبکه آدالاین شبیه پرسپترون است ولی با تابع تبدیل خطی (به جای آستانه دو مقداره)



$$\underline{a} = W\,\underline{p} + \underline{b}$$

معادلات ویدرو-هوف در حالت تک نرون

- بازنویسی معادله خروجی با فرمول بندی جدید

$$\underline{x} = \begin{bmatrix} W_{1,1} \\ \vdots \\ W_{1,r} \\ b \end{bmatrix} = \begin{bmatrix} W_1^T \\ b \end{bmatrix} \quad \underline{q} = \begin{bmatrix} \underline{p} \\ 1 \end{bmatrix}$$

$$a = \underline{x}^T \underline{q}$$

- تابع هزینه میانگین مربعات خطا

$$e = t - a = t - \underline{x}^T \underline{q}$$

$$F(\underline{x}) = E(e^2) = \underline{x}^T R \underline{x} - 2\underline{x}^T \underline{d} + c$$

- تعیین نقطه ایستا از روی گرادیان تابع

$$\nabla F(\underline{x}) = -2\underline{d} + 2R\underline{x}$$

$$\nabla F(\underline{x}) = 0 \Rightarrow \underline{x}^* = R^{-1}\underline{d}$$

توجه

- اگر بتوانیم معکوس ماتریس را محاسبه کنیم نیازی به الگوریتمهای مینیمم سازی نداریم.
- اگر نخواهیم (یا نتوانیم) معکوس R را محاسبه کنیم، الگوریتم مینیمم سازی SD را می توان به کار برد.
  - در این حالت نیاز به محاسبه گرادیان تابع داریم.
- در حالت کلی مطلوب یا مناسب نیست که بردار d و ماتریس R محاسبه شوند. لذا تقریبی از الگوریتم SD یا همان LMS استفاده می شود.
  - استفاده از گرادیان لحظه ای به جای گرادیان واقعی

الگوریتم LMS

- در واقع استفاده از خطای لحظه ای به عنوان شاخص عملکرد

$$\hat{F}(\underline{x}) = e^2(k) = (t(k) - a(k))^2$$

$$\nabla\hat{F}(\underline{x}) \equiv \hat{\nabla}F(\underline{x}) = \nabla e^2(k)$$

- دقت شود که $\nabla F(\underline{x}) = E[\nabla\hat{F}(\underline{x})]$
- به کمک قاعده زنجیره ای به سادگی دیده می شود که $\nabla\hat{F}(\underline{x}) = -2e(k)\underline{q}(k)$
- قانون LMS در حالت تک نرون:

$$\underline{x}(k+1) = \underline{x}(k) + 2\alpha e(k)\underline{q}(k)$$

- قانون LMS در حالت کلی:

$$W(k+1) = W(k) + 2\alpha\underline{e}(k)(\underline{p}(k))^T$$

$$\underline{b}(k+1) = \underline{b}(k) + 2\alpha\underline{e}(k)$$

فرم دسته ای یادگیری LMS در حالت تک نرون

- شاخص اجرایی در حالت تک نرون
  - N – تعداد داده های یادگیری

$$\bar{F}(\underline{x}) = \frac{1}{N}\sum_{k=1}^{N} e^2(k)$$

- شاخص اجرایی در حالت کلی S نرون

$$\bar{F}(\underline{x}) = \frac{1}{N}\sum_{k=1}^{N} \underline{e}^T(k)\underline{e}(k)$$

- در حالت تک نرون به سادگی دیده می شود:

$$\nabla\bar{F}(\underline{x}) = \frac{-2}{N}\sum_{k=1}^{N} e(k)\underline{q}(k)$$

$$\underline{x}(k+1) = \underline{x}(k) + \frac{2\alpha}{N}\sum_{k=1}^{N} e(k)\underline{q}(k)$$

فرم دسته ای یادگیری LMS در حالت کلی

$$W(k+1)=W(k)+\frac{2\alpha}{N}\begin{bmatrix}\sum_{k=1}^{N}e_1(k)(\underline{p}(k))^T\\ \vdots \\ \sum_{k=1}^{N}e_S(k)(\underline{p}(k))^T\end{bmatrix}=W(k)+\frac{2\alpha}{N}\sum_{k=1}^{N}\underline{e}(k)(\underline{p}(k))^T$$

$$\underline{b}(k+1)=\underline{b}(k)+\frac{2\alpha}{N}\begin{bmatrix}\sum_{k=1}^{N}e_1(k)\\ \vdots \\ \sum_{k=1}^{N}e_S(k)\end{bmatrix}=\underline{b}(k)+\frac{2\alpha}{N}\sum_{k=1}^{N}\underline{e}(k)$$

- تمرین ۱: شبکه آدالاین برای مثال سیب پرتقال گلابی در هر دو حالت ترتیبی و دسته ای شبیه سازی و نتایج مقایسه شود. (بخش ۷-۶-۲- کتاب)
- تمرین ۲: مسئله XOR توسط شبکه آدالاین شود. (دقت شود که پرسپترون تک لایه قادر به حل این مسئله نیست)

بهبود هایی بر LMS (مطالعه آزاد، بخش ۷-۸)

- نرخ یادگیری متغیر با زمان
- استفاده از مومنتوم

کاربرد شبکه آدالاین در فیلترهای تطبیقی

- کاربردهای عملی فراوان در پردازش سیگنال دیجیتال
- استفاده از بلوک TDL (Tapped Delay Line) در ورودی شبکه
- حذف تطبیقی نویز (بررسی در کلاس حل تمرین)