دانشگاه تهران

دانشکده الگوریتم و محاسبات

استاد درس:

دکتر شعبان‌خواه

گزارش دهنده:

مصطفی محمدی قراسویی

۸۱۰۸۹۷۰۳۰

درس:

Fundamental of Neural Networks

تمرین:

TSP by SOM

شرح

در این تمرین می‌خوهیم یک مثال از کلاسترین که به روش linear topology می‌باشد و از SOM برای حل مسأله بهینه سازی محدود شده استفاده می‌کند را بکار ببریم. این تمرین سعی در حل مسأله فروشنده دوره‌گرد را دارد. مشکل فروشنده دوره‌گرد به این صورت است که می‌خواهد با حداقل سفر ممکن به تمامی شهرهای انتخابی سفر کند به گونه‌ای که هر شهر را فقط یک بار بازدید کند و در نهایت به شهر مبدا برگردد.

برای حل این تمرین ما از الگوریتم unsupervised کلاسترینگ SOM استفاده خواهیم کرد به این گونه که این الگوریتم در هر بار اجرای خود در یک حلقه که ۱۰۰ مرتبه اجرا می‌شود، به دنبال کوتاهترین مسیرها می‌گردد. جستجوی کوتاهترین مسیر به این صورت است که به تعداد شهرهای انتخابی کلاستر داریم. هنگامی که کلاستری منتخب به روزرسانی می‌شود کلاسترهای اطراف خود را نیز به روزرسانی می‌کند. این الگوریتم برای یافتن کوتاهترین مربع فاصله از الگوریتم کمترین فاصله اقلیدوسی استفاده می‌کند. در این الگوریتم به‌روزرسانی هر کلاستر از روش linear است و پس از گذشت چند دور از اجرای حلقه داخلی، کلاستر منتخب دیگر کلاسترهای همسایه خود را مجاز به به‌روز رسانی نمیداند و تنها خود کلاستر به روزرسانی می‌شود. لازم به ذکر است که الگوریتم از پارامتر learning rate نیز استفاده می‌کند که در طی اجرای الگوریتم در حلقه اصلی این پارامتر مقدارش رفته رفته کاهش می‌یابد.

الگوریتم در بررسی‌های مختلف انجام شده هنگام تست به این نتیجه رسیده شده که انتخاب مناسب ضریب کاهش learning rate(alpha) در تشخیص درست و کارایی الگوریتم بسیار حائز اهمیت است به گونه‌ای که اگر alpha مناسب انتخاب نشود حتی الگوریتم می‌تواند جواب نادرستی تولید کند و یافتن کوتاه‌ترین مسیر با مشکل مواجه شود. از طرفی دیگر اگر مقدار radius (روش به روز رسانی کلاسترهای مختلف) نیز درست انجام نشود یا به عبارت دیگر از روش no topology استفاده شود باز هم الگوریتم در کارایی خود به منظور پیدا کردن کوتاهترین مسیر با مشکل مواجه خواهد شد.

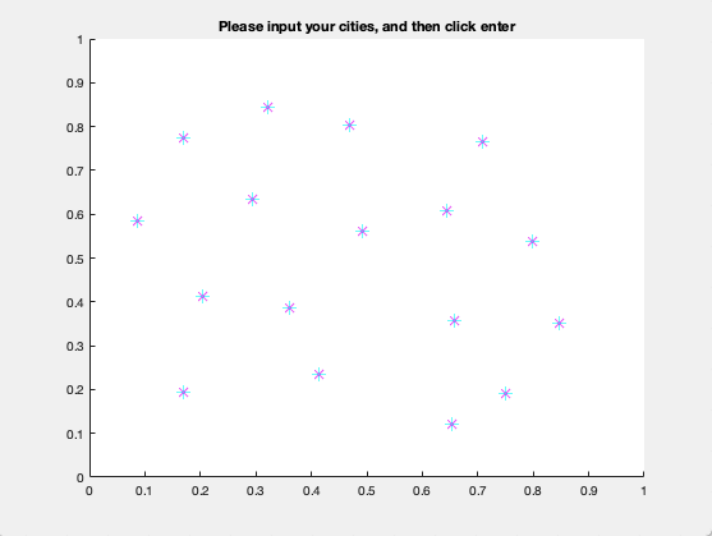
شبه کد مربوط به پیاده سازی الگوریتم TSP همان شبه کد معروف الگوریتم SOM است.

برنامه‌های نوشته شده با استفاده از Matlab بوده و به دو صورت پیاده سازی شده است :

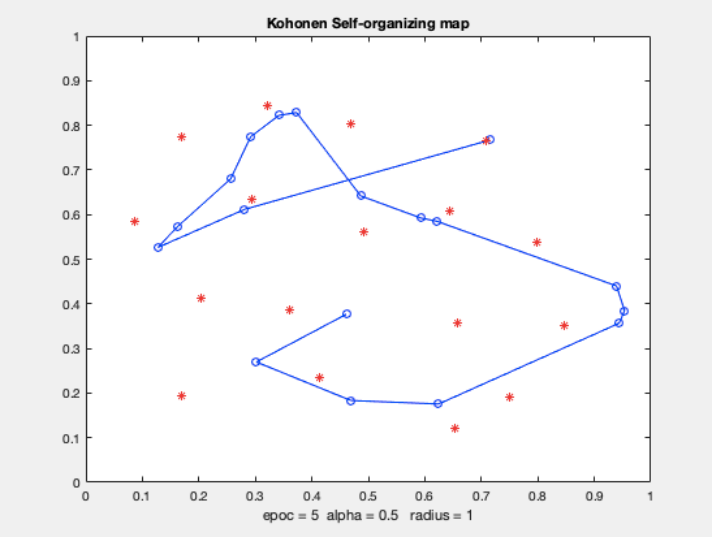
۱- ورود و انتخاب مختصات شهرها توسط کاربر از طریق کلیک کردن ماوس روی صفحه plot اتفاق می‌افتد و هنگامی که آخرین شهر را وارد شد با زدن دکمه enter الگوریتم کار خود را پیش می‌برد. در این الگوریتم محدودیتی در تعداد ورود شهرها نداریم و انتخاب شهرها پس از ورود آن به صورت اتوماتیک توسط برنامه تشخیص داده شده و کلاسترهای مربوطه شروع به ارزیابی فاصله اقلیدوسی خود می‌کنند. کار با تکرار ۱۰۰ مرحله در این برنامه انجام می‌شود. ۵۰ تکرار اول با radius ۱ و ۵۰ تکرار دوم با radius ۰ اجرا می‌شود. در این ۱۰۰ تکرار ضریب یادگیری ماشین رفته رفته کاهش

می‌یابد به طوری که از ضریب یادگیری ۰.۵ به ضریب یادگیری ۰.۳۲۸۰۵ می رسیم. نام این برنامه TSP_with_manual_Cities.m است.

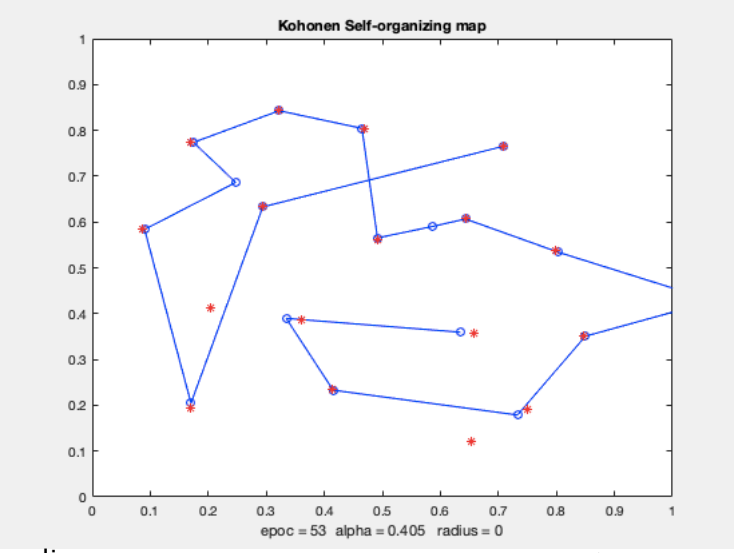
توضیحات بخش‌های مختلف کد در خود برنامه نوشته شده است و به همین خاطر از بیان دوباره مطالب صرف نظر می‌شود. در زیر چند نمونه از بخش‌های مختلف اجرا قرارداده می‌شود.



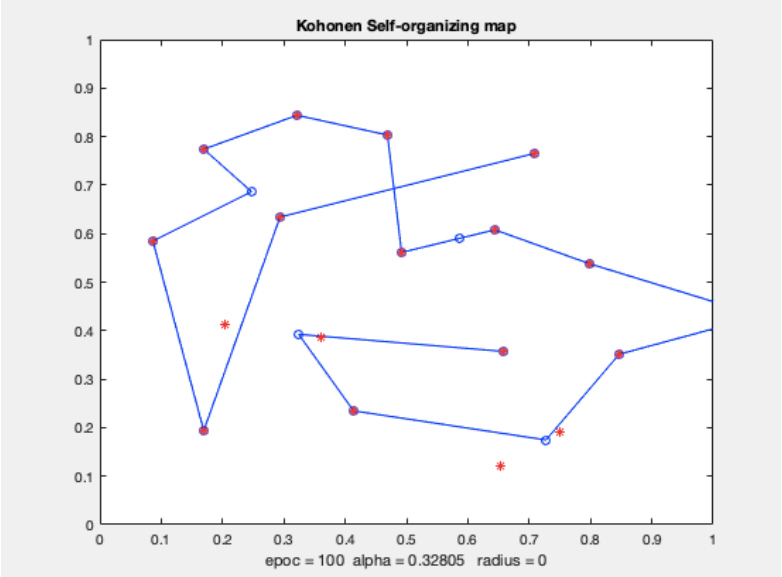
شکل ۱: نمایش پنجره plot به منظور ورود شهرها در مختصات دلخواه توسط کاربر



شکل ۲: اجرای الگوریتم در epoc پنجم با ضریب یادگیری ۰.۵ و radius ۱

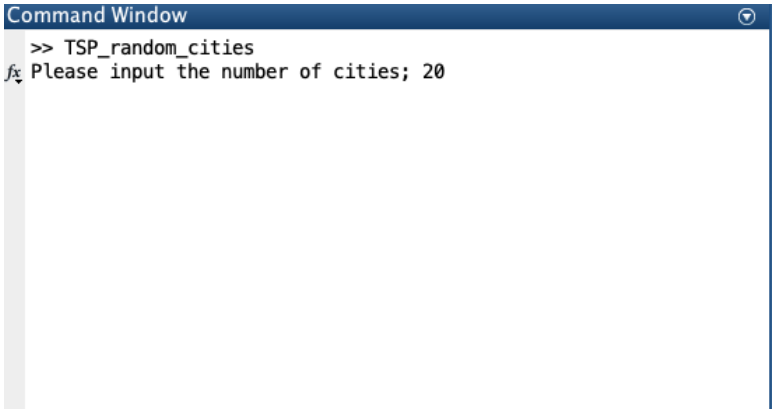


شکل ۳: اجرای الگوریتم در epoc پنجاه و سوم با ضریب یادگیری ۰.۴۰۵ و radius ۰

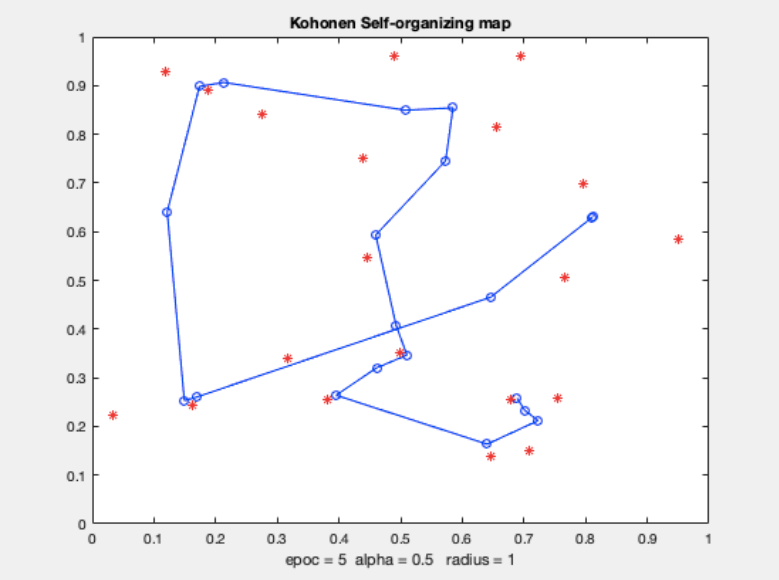


شکل ۴: اجرای الگوریتم در epoc صدم با ضریب یادگیری ۰.۳۲۸۰۵ و radius ۰

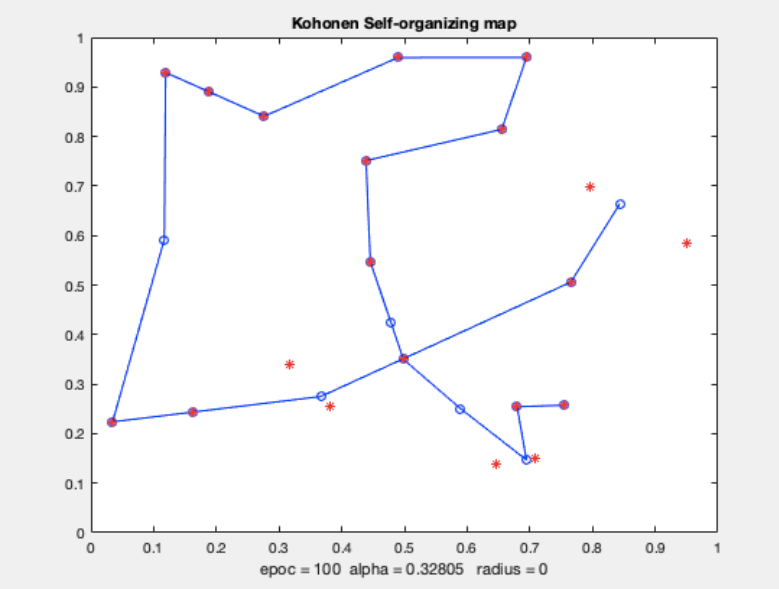
۲- ورود شهرها. با. استفاده از صفحه کلید اتفاق می‌افتد ولی انتخاب مختصات شهرها. توسط. خود الگوریتم به. صورت کاملا تصادفی انجام می‌شود. این برنامه دقیقا شبیه برنامه شماره ۱ است و تفاوت آن تنها در وارد کردن شهرها است. نام این برنامه TSP_random_cities.m است.



شکل ۵: اجرای الگوریتم و پرسش از کاربر برای ورود تعداد شهرها



شکل ۶: اجرای الگوریتم در مرحله پنجم با raius ۱ و alpha ۰.۵



شکل ۷: اجرای الگوریتم در مرحله صدم.

پایان گزارش