جمهوری اسلامی ایران

دانشگاه صنعتی شاهرود

گزارش نهایی طرح پژوهشی
کد ۱۱۰۴

امکان استفاده از هندسه فراکتال در پردازش داده‌های نقشه‌برداری

مجری

بهزاد تخم‌چی

دانشکده معدن و ژئوفیزیک

بسم الله الرحمن الرحیم

پیشگفتار

گزارش حاضر نتیجه مطالعات صورت گرفته در راستای انجام طرح پژوهشی با عنوان *امکان استفاده از هندسه فراکتال در پردازش داده‌های نقشه‌برداری* به کد ۱۱۰٤ مصوبه شورای محترم پژوهشی دانشگاه صنعتی شاهرود می‌باشد که در تاریخ ۱۳۸۲/۹/۱٦ به تصویب یکصد و یازدهمین جلسه شورای پژوهشی دانشگاه رسیده است. بدینوسیله از مساعدتهای مسئولین محترم دانشگاه و بالاخص مدیران محترم پژوهشی و سرکار خانم آزادخواه کمال تشکر بعمل می‌آید.

در انجام مطالعه حاضر از راهنمائیهای اساتید محترم، آقایان دکتر مهدی ایران نژاد و دکتر رضا خالوکاکائی بهره‌ها برده‌ام که از ایشان صمیمانه تقدیر می‌نمایم. به انضمام گزارش، نرم‌افزارهایی ارائه گشته که حاصل زحمات آقای سید علی کاظمی دانشجوی محترم رشته اکتشاف معدن می‌باشد، البته در انجام پروژه نیز از همفکریهای ایشان بهره فراوان برده‌ام. برایشان آرزوی توفیق می‌کنم. همچنین از خانم‌ها عصمت افشارنیا و نرگس بهی و آقایان سید بیژن ماهباز و سید علی‌هاشمی نسب که بنده را در انجام طرح یاری رسانده‌اند تشکر می‌کنم.

در نهایت امیدوارم نتایج طرح مورد استفاده همکاران محترم دانشگاهی قرار بگیرد.

بهزاد تخم چی

فهرست مطالب

عنوان	صفحه
فصل یک: کلیات	
۱-۱ مقدمه	۱
۱-۲ روش‌های پردازش اطلاعات	۲
۱-۳ خطای تخمین	۳
فصل دوم: کلیاتی راجع به فراکتال‌ها	
۲-۱ مقدمه	۴
۲-۲ شکل سطوح طبیعی	۵
۲-۳ بعد فراکتال	۶
۲-۴ کاربرد بعد فراکتال	۷
۲-۵ کاربرد فراکتال در پردازش اطلاعات توپوگرافی	۷
فصل سوم: شرح عملیات نقشه‌برداری	
۳-۱ مقدمه	۹
۳-۲ برداشت توسط RDS	۱۰
۳-۳ برداشت توسط GPS	۱۰
۳-۴ پردازش اطلاعات	۱۱
فصل چهارم: روشهای محاسبه بعد فراکتال	
۴-۱ مقدمه	۱۲
۴-۲ روش پرگار تقسیم سیار	۱۲
۴-۳ روش پرگار تقسیم سیار دو بعدی	۱۴
۴-۴ استفاده از تغییر نما	۱۵

عنوان **صفحه**

۴-۵ ارائه نرم‌افزار ۱۶

۴-۵-۱ شرح عملکرد نرم‌افزار ۱۶

۴-۵-۲ نحوه خواندن اطلاعات ۱۷

۴-۵-۳ محاسبه بعد فراکتال به روش پرگار تقسیم سیار ۱۷

۴-۵-۴ محاسبه بعد فراکتال به روش تغییرنما ۲۰

۴-۵-۵ محاسبه بعد فراکتال به روش پرگار تقسیم سیار دو بعدی ۲۱

فصل پنجم: تولید داده‌ای معتبر آماری با بعد فراکتال معین

۵-۱ مقدمه ۲۳

۵-۲ روشهای تولید نویز (آشفتگی) براونی ۲۳

۵-۳ الگوریتم تولید حرکت براونی کسری یک بعدی ۲۴

۵-۳-۱ روش اول تولید حرکت براونی کسری یک بعدی ۲۴

۵-۳-۲ روش دوم تولید حرکت براونی کسری یک بعدی ۲۴

۵-۴ حرکت براونی کسری دو بعدی ۲۵

۵-۵ نحوه خواندن اطلاعات ۲۷

۵-۶ ساختن ماتریس اعداد تصادفی گوسی ۳۰

۵-۷ ساختن ماتریس اعداد نویز براونی توسط الگوریتم تولید حرکت براونی کسری ۳۰

۵-۸ محاسبه بعد فراکتال به روش تغییرنما ۳۰

۵-۹ اجرای الگوریتم جابجایی نقطه میانی ۳۰

۵-۱۰ نحوه ذخیره محاسبات ۳۲

عنوان | صفحه

فصل ششم: نحوه کار با نرم‌افزار FM و FS

۶-۱ مقدمه ۳۳

۶-۲ نحوه ارسال داده‌ها به نرم‌افزارهای FM و FS ۳۳

۶-۳ طریقه کلی استفاده از نرم‌افزار FM ۳۷

۶-۳-۱ نحوه محاسبه بعد فراکتال به روش پرگار تقسیم سیار ۳۷

۶-۳-۲ محاسبه بعد فراکتال به روش تغییرنما ۴۰

۶-۳-۳ محاسبه بعد فراکتال به روش پرگار تقسیم سیار دو بعدی ۴۲

۶-۴ نحوه استفاده ازنرم‌افزار Fractal Simulation ۴۳

۶-۵ توضیحات تکمیلی درباره نرم‌افزار (FS) Fractal Simulation ۴۵

فصل هفتم: نتیجه‌گیری و پیشنهادات

۷-۱ نتیجه‌گیری ۵۰

۷-۱ پیشنهادات ۵۱

پیوست اول: کدهای اصلی فرم اصلی نرم‌افزار (FM) Fractal Methods ۵۲

پیوست دوم: کدهای اصلی فرم نمایش گراف نرم‌افزار (FM) Fractal Methods ۶۳

پیوست سوم: کدهای اصلی فرم اصلی نرم‌افزار (FS) Fractal Simulation ۶۷

فهرست منابع به ترتیب استفاده در متن ۸۳

فهرست اشکال

عنوان **صفحه**

شکل ۱-۱ مثالی از خطای فاحش تخمینگر خطی ۳

شکل۲-۱ خط ساحلی انگلستان با گام‌های مختلف پیموده شده است ۶

شکل ۲-۲ نحوه بدست آوردن شیب فراکتال خط ساحلی انگلستان ۷

شکل ۴-۱ شکل شماتیک جهت نمایش لگاریتم طول خط کش در مقابل لگاریتم طول منحنی ۱۳

شکل ۴-۲ مقایسه روش‌های *الف* و ب پرگار تقسیم سیار در محاسبه بعد فراکتال ۱۴

شکل ۴-۳ یک شبکه نمونه جهت محاسبه مقادیر تغییرنما در گامهای مختلف ۱۶

شکل ۴-۴ شمای عملیاتی زیرروالهای برنامه FM ۱۸

شکل ۴-۵ جزئیات عملکرد فرمت ب روش پرگار تقسیم سیار ۱۹

شکل ۴-۶ نمایشی از مفاهیم مساحت کلی سطح و مساحت گام در روش پرگار تقسیم سیار دو بعدی ۲۲

شکل ۵-۱ حرکت براونی با پارازیت‌های سفید آن (پیتجن و سوپ ۱۹۸۸) ۲۵

شکل ۵-۲ مثال تولید اعداد تصادفی حرکت و براونی کسری برای دو مقدار متفاوت از H که
توسط نرم افزار تهیه شده رسم شده است ۲۶

شکل ۵-۳ جابجایی نقطه میانی برای حرکت براونی یک بعدی ۲۶

شکل ۵-۴ سه مرحله اول الگوریتم جابجایی نقطه میانی بر روی یک شبکه منظم ۲۸

شکل ۵-۵ شمای عملیاتی زیر روالهای برنامه شبیه‌ساز فراکتال (FS) و ارتباط زیر روالها با کلاس مادر ۲۹

شکل ۵-۶ انواع حالات ممکن به لحاظ مجاورت داده معتبر با داده‌های بلانک شده و
رفتار نرم افزار با هر کدام از این حالات ۳۱

شکل ۶-۱ شمای گرافیکی پنجره Scattered data interpolation ۳۴

شکل ۶-۲ شمای گرافیکی پنجره Save Grid As ۳۵

شکل ۶-۳ نمایش نحوه انتخاب دستور Sort از منوی Data ۳۶

عنوان **صفحه**

شکل ۶-۴ نحوه تنظیمات پنجره Sort ۳۶

شکل ۶-۵ تنظیمات پنجره Gsi ۳۷

شکل ۶-۶ شمای ظاهری برنامه قبل از ورود اطلاعات ۳۸

شکل ۶-۷ شمای ظاهری برنامه پس از ورود اطلاعات ۳۹

شکل ۶-۸ نحوه انتخاب دستورات مختلف محاسبه و نمایش مقادیر بعد فراکتال ۴۰

شکل ۶-۹ پنجره Loop Number ۴۰

شکل ۶-۱۰ نمایی از پنجره Graph ۴۱

شکل ۶-۱۱ نحوه ذخیره کردن نتایج ۴۲

شکل ۶-۱۲ نمایش مقدار تغییرنمای سطح توسط که به توسط نرم‌افزار محاسبه شده است ۴۲

شکل ۶-۱۳ شمایی از کادر Random number generator scale ۴۳

شکل ۶-۱۴ نقشه توپوگرافی و سه‌بعدی حاصل از کریجینگ داده‌های منطقه محلات ۴

با فواصل خطوط شبکه ۴۰ متر ۴۶

شکل ۶-۱۵ نقشه توپوگرافی و سه‌بعدی حاصل از جابجایی نقطه میانی داده‌های منطقه محلات ۴

با فواصل خطوط شبکه ۲۰ متر ۴۷

شکل ۶-۱۶ نقشه توپوگرافی و سه‌بعدی حاصل از جابجایی نقطه میانی داده‌های منطقه محلات ۴

با فواصل خطوط شبکه ۱۰ متر ۴۸

شکل ۶-۱۷ نقشه توپوگرافی و سه‌بعدی حاصل از کریجینگ داده‌های منطقه محلات ۴

با فواصل خطوط شبکه ۱۰ متر ۴۹

فهرست جداول

عنوان *صفحه*

جدول ۲-۱ طول ساحل انگلیس با مقیاس‌های مختلف ۴
جدول ۴-۱ شمای ماتریس ذخیره نتایج در روش پرگار تقسیم سیار ۱۷
جدول ۴-۲ ماتریس ذخیره نتایج در روش پرگار تقسیم سیار دو بعدی ۲۱
جدول ۶-۱ نتایج چگال کردن سطوح و بعد فراکتال حاصل از آنها ۴۵

چکیده

در هندسه فراکتال بر خلاف هندسه اقلیدسی بحث از خطوط و سطوحی است که بعد عدد صحیح یک و دو ندارند. در این تحقیق منطق و نرم‌افزاری جهت محاسبه بعد فراکتال سطوح طبیعی هنگام نقشه‌برداری ارائه گردیده و سپس تخمینگری با منطق فراکتال به نام جابجایی نقطه میانی جهت چگال کردن اطلاعات ارائه شده است. نحوه عمل تخمینگر بگونه‌ای است که در نهایت بعد فراکتال نقشه‌ای که بعنوان نقشه توپوگرافی سطح ترسیم می‌شود برابر با بعد فراکتال سطح اولیه خواهد بود. بعد فراکتال خطوط و سطوح به سه روش پرگار تقسیم سیار، پرگار تقسیم سیار دوبعدی و تغییرنما محاسبه می‌شود.

در نهایت اثبات شده که سطوح شبیه‌سازی شده به روش فراکتال به لحاظ آماری کاملاً شبیه به سطوح اولیه بوده و از این منظر تخمینگر فراکتال نسبت به تخمینگرهای دیگر قابل اعتمادتر است.

۱

کلیات

۱-۱ مقدمه

تهیه نقشه دقیق و استفاده صحیح از آن در طرح و اجرای هر پروژه‌ای یکی از عوامل موفقیت آن پروژه به شمار می‌آید. برای تحقق این امر متخصصین نقشه‌برداری همواره کوشش می‌نمایند تا با بهره‌گیری از وسایل جدید، روش‌های مناسب پردازش اطلاعات و شیوه‌های مدرن ترسیم دقت نقشه‌ها را بیش از پیش افزایش دهند.

به طور کلی نقشه‌برداری تلفیقی از سه فعالیت برداشت‌های صحرایی توسط وسایل نقشه‌برداری، ریاضیات کاربردی جهت پردازش داده‌های برداشت شده و هنر ترسیم نقشه می‌باشد. بدیهی است که در تمامی مراحل عملیاتی مذکور، میزانی از خطا نهفته است. به عنوان مثال برداشت‌های صحرایی به وسیله دستگاه‌های نقشه‌برداری که توسط انسان ساخته شده‌اند صورت می‌گیرد. از آنجا که عملکرد انسان همراه با خطا است، بدون شک میزانی خطای سیستماتیک در هنگام ساخت دستگاه‌ها وجود خواهد داشت که تأثیر خود را در توأم با خطا بودن اطلاعات اولیه خواهد گذاشت. در ضمن کاربر نیز انسان است که فعالیتش توأم با خطا است و تفسیر شرایط طبیعی نیز خطا وارد مجموعه عملیات خواهد نمود. بنابراین در مرحله برداشت سه نوع خطای دستگاهی، انسانی و طبیعی وجود خواهد داشت.

به همین ترتیب در پردازش اطلاعات و ترسیم نقشه نیز خطا وجود خواهد داشت که هنر نقشه‌بردار، کاهش مجموع خطاها و نزدیک کردن هر چه بیشتر نقشه تهیه شده به واقعیت است.

۱-۲ روش‌های پردازش اطلاعات

پردازش اطلاعات نقشه‌برداری شامل دو مرحله است. ابتدا توسط ریاضیات کاربردی می‌بایستی اطلاعات خام اولیه را به مختصات دکارتی تبدیل نمود. برای این منظور بسته به اینکه از چه نوع دستگاهی جهت برداشت اطلاعات اولیه استفاده شده باشد، روابط خاصی موجود است. بنابراین عموماً در انجام این مرحله از پردازش، مشکل و یا خطای خاصی وجود نخواهد داشت.

در مرحله دوم با توجه به اینکه نسبت اطلاعات به مجهولات پائین است، باید به توسط یک تخمینگر، برای بعضی از نقاط مجهول، ارتفاع را تخمین زد. بدیهی است از تخمین‌گرهای دستی و یا کامپیوتری می‌توان استفاده کرد. اما با توجه به حجم بالای محاسبات ریاضی تنها تخمین‌گر دستی، درونیابی ـ برونیابی است. در این شیوه فرض بر این است که تغییرات ارتفاعی بین دو نقطه به صورت خطی است. بنابراین به عنوان مثال دقیقاً بین دو نقطه به فاصله یکصد متر از هم که ارتفاع آنها به ترتیب صفر و صد متر است، ارتفاع پنجاه متر خواهد بود. اما در روشهای نرم‌افزاری از تخمینگرهای متفاوتی می‌توان استفاده کرد. به عنوان مثال در نرم‌افزار سورفر که یک نرم‌افزار نقشه‌برداری است، هشت تخمینگر ارائه شده است. این تخمینگرها عبارتند از: عکس مجذور فاصله[1]، کریجینگ[2]، کمینه انحناء[3]، نزدیکترین همسایه[4]، رگرسیون چندگانه[5]، توابع پایه شعاعی[6]، روش شپارد[7] و درونیابی خطی[8] هستند.

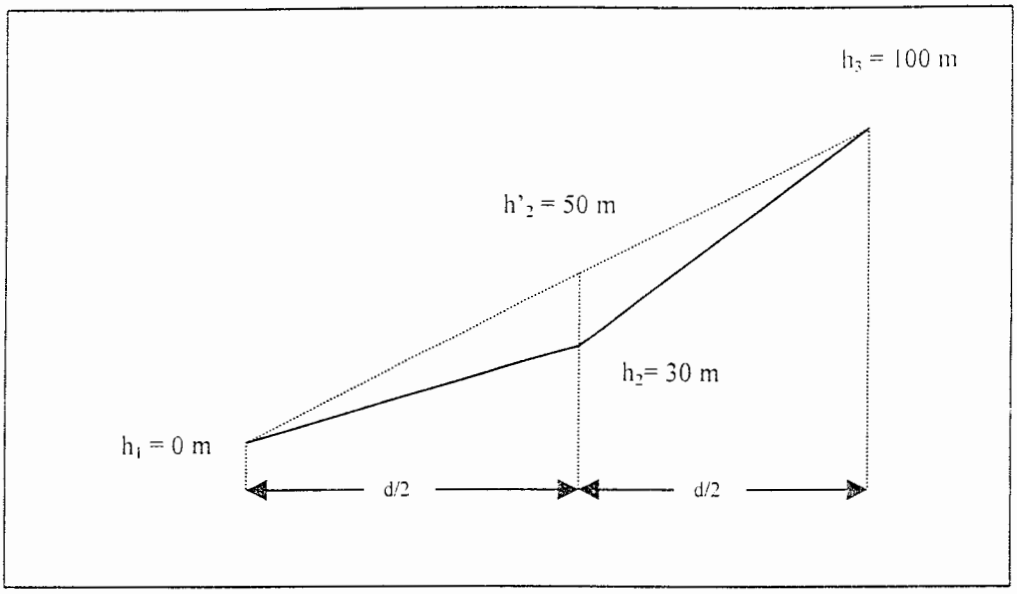
هر کدام از تخمینگر، در بعضی از مورفولوژی‌های خاص کاربرد دارند. به عنوان مثال، احتمالاً در مورفولوژی تبخیری، روش‌های عکس مجذور فاصله و شپارد و در مورفولوژی هموار، روشهای کریجینگ و کمینه انحناء تخمین بهتری از سطح ارائه خواهند نمود.

۱-۳ خطای تخمین

تخمینگر خوب، تخمینگری است که ارتفاع نقاط مجهول را هرچه نزدیکتر به مقدار واقعی تخمین بزند. به عبارت دیگر، احتمالاً مقادیر تخمین زده شده با مقادیر واقعی برابر نیستند. جالب اینجاست که بعضاً این اختلاف یا خطا می‌تواند بسیار بزرگ هم باشد. به عنوان مثال در شکل ۱-۱ به فرض نقطه وسط مجهول می‌بود. در این صورت تخمینگر خطی، تخمین خط چین را ارائه می‌نمود. به

1- Inverse Distance
2- Kriging
3- Minimum Curvature
4- Nearest Neighbor
5- Polynomial Regression
6- Radial Basis Functions
7- Shepards Method
8- Triangulation With Linear Interpolation

عبارت دیگر برای نقطه وسط، ارتفاعی معادل ۵۰ متر تخمین زده می‌شد. حال آنکه، همچنانکه در شکل مشاهده می‌شود، ارتفاع نقطه وسط معادل ۳۰ متر است. بنابراین خطای تخمین ۲۰ متر خواهد بود که بدون شک در نقشه‌های با مقیاس بزرگ مورد قبول نیست.



شکل ۱-۱ مثالی از خطای فاحش تخمینگر خطی

بدون شک، هر شخصی با اندکی تفکر مثال‌های مشابهی می‌تواند ارائه دهد که تخمینگرهای غیرخطی نیز خطای فاحشی داشته باشند. بنابراین ملاحظه می‌شود هر چند که توانمندی دستگاهها بیشتر شده و خطاهای برداشت و ترسیم کاهش یافته، کاستن از خطای پردازش ضروری است. البته به عنوان یک راهکار جهت کاستن از خطای پردازش می‌توان فاصله بین نقاط برداشت را خیلی کاهش داد، اما بدیهی است که این روش بدلیل اینکه هزینه‌های عملیات صحرایی و زمان اجرایی عملیات افزایش می‌یابد، مطلوب نیست. تخمین گرهایی نیز همچون کریجینگ وجود دارند که در صورت چگال بودن شبکه برداشت، می‌توانند ارتباط فضایی بین نقاط اطلاعاتی را کشف کرده و نتیجتاً به عنوان یک تخمینگر با حداقل واریانس تخمین عمل می‌نماید. اما از ویژگی‌های این تخمینگر این است که نرم‌ترین منحنی را بر روی شبکه منطبق می‌کند، در صورتی که بعضی مواقع در طبیعت مورفولوژی‌های زبر نیز وجود دارند که بدون شک تخمینگرهای نرم کننده برای آن مناطق مناسب نیستند.

در فصول آتی تخمینگری معرفی خواهد گشت که امکان ساختن سطوح زبر توسط آن وجود دارد.

چنانکه در جدول ۲-۱ ملاحظه می‌شود، همواره با کوچک شدن واحد اندازه‌گیری، طولی که از این اندازه‌گیری‌ها حاصل می‌شود بزرگتر می‌شود. در پایان از آنجا که محدودیتی برای کوچک شدن واحد وجود ندارد، با بینهایت بودن طول این خط روبرو می‌شویم. ماندلبروت در جستجوی پاسخی برای این پرسش نظریات تازه‌ای را مطرح ساخت که به تحولی فراگیر در درک و تصوری که تا کنون از بعد وجود داشت انجامید و منجر به پیدایش هندسه فراکتال شد. وی تاکید نمود که با توجه به متغییر بودن مقدار طول نامبرده، دیگر سخن گفتن از طول چنین شکلهای بی‌قائده‌ای بیهوده است. وی همه پدیده‌هایی را که با تعاریف معمول با نقطه، خط، صفحه و پیکر هندسی نمی‌توان توصیف کرد از جمله این اشکال دانست [۱].

همانطور که می‌دانیم در هندسه رایج(اقلیدسی) فقط ابعاد عدد صحیح وجود دارد: خط دارای یک بعد، صفحه دو بعد و مکعب دارای سه بعد است. حال آنکه برای هر خط شکسته یا سطح ناهمواری می‌توان بعدی را بدست آورد که عدد صحیح نیست [۲].

۲-۲ شکل سطوح طبیعی

بیشتر مردم روی خشکی زندگی می‌کنند. بر روی زمین راه رفته و بر روی تخته سنگ می‌نشینند. اما عموماً به اشکال مختلف سطوح مشترکی که به طور روزمره با آنها برخورد دارند، توجه نمی‌کنند. بنابراین جای تعجب خواهد بود اگر بدانند شکلهایی که رؤیت می‌کنند، بسته به مقیاس مشاهده متفاوت هستند. به عنوان مثال یک فضانورد از فضا، زمین را به شکل یک توپ آرام و هموار مشاهده می‌کند، در حالیکه هنگام بالا رفتن از یک کوه، شکل زمین به هر چیز جز یک توپ آرام و هموار شبیه است. از این مباحث می‌توان به یک نتیجه دست پیدا کرد: ممکن است سطوح خارجی، همچون کوه‌های هیمالیا هموار باشند. البته این امر با نگاهی از فضا اتفاق می‌افتد. اما همان سطوح با مشاهده از روی زمین کاملاً ناهموار هستند [۲].

به طور کلی شکل ظاهری به میزان فاصله مشاهده شده بستگی دارد. بنابراین این سؤال مطرح می‌گردد: چگونه می‌توان شکل پدیده‌ای را توصیف کرد که با چشم هموار، ولی به وسیله میکروسکوپ ناهموار به نظر می‌رسد [۳]؟

همچون سؤال بالا را می‌توان در مورد سطوح طبیعی زمین‌شناسی مطرح ساخت. همچنانکه می‌دانیم بعضی از سطوح در اثر فرایند رسوب‌گذاری به وجود می‌آیند. بعضی دیگر در اثر فرسایش یا پوسیدگی بوجود می‌آیند. بعضی نیز به دلیل محیط رشد ناهمگون توسعه پیدا می‌کنند [۳]. بدیهی است که مورفولوژی این سطوح با هم متفاوت بوده و اصولاً شکل متفاوت با هم دارند. بنابراین می‌توان بیان نمود که بسیاری از ساختارهای زمین‌شناسی همچون توپوگرافی از آمار فراکتالی پیروی می‌کنند.

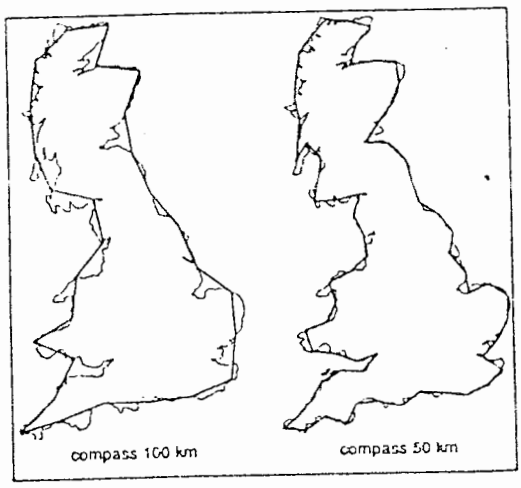
جهت مطالعه ساختارهای فراکتالی می‌بایست بعد فراکتال ساختارها را اندازه‌گیری و محاسبه کرد. بدین ترتیب ابزار کمی بدست خواهد آمد که توسط آن می‌توان بین شکل ساختارهای مختلف

تمایز قائل شد. به عنوان مثال یک خط مستقیم بعد فراکتالی یک داشته، اما یک خط شکسته یا منحنی بعد فراکتالی بزرگتر از یک دارد. به همچنین یک سطح صاف و ملایم بعد فراکتالی دو داشته، اما یک سطح با تغییرات ارتفاعی شدید، بعد فراکتالی بیش از دو خواهد داشت. در ادامه مطالب بیشتری راجع به فراکتال ارائه می‌شود.

۲-۳ بعد فراکتال

به روش‌های مختلفی می‌توان بعد به اصطلاح فراکتالی را محاسبه کرد که ساده‌ترین آن چنین است [۱]:

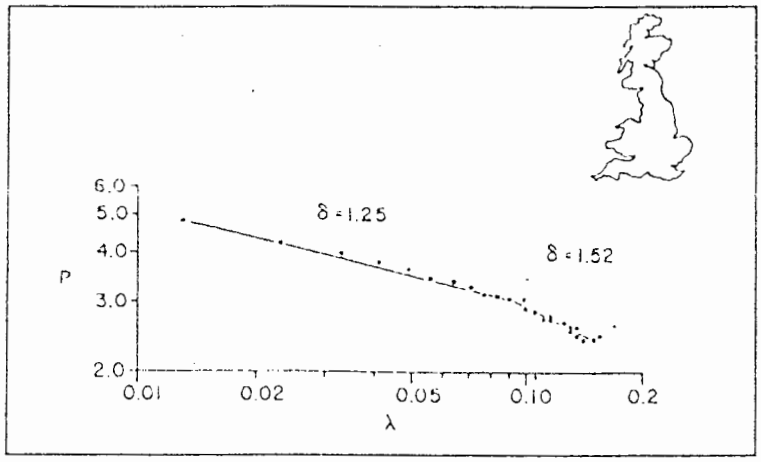
برای اندازه گرفتن طول یک خط بر روی نقشه، ابتدا به کمک پرگار طول معینی(r) را به عنوان مقیاس در نظر گرفته، خط را می‌پیمائیم. طول کامل این خط(L) از ضرب تعداد گام‌ها(N) در مقیاس(r) بدست می‌آید($L=N.r$). بدیهی است مقدار باقی‌مانده را نیز با این حاصلضرب جمع می‌نمائیم. عمل یاد شده در مقیاس‌های مختلف(r) تکرار می‌شود. برای مثال در شکل ۲-۱ دو مرحله از انجام این عملیات بر روی سواحل انگلستان آورده شده است.



شکل۲-۱ خط ساحلی انگلستان با گام‌های مختلف پیموده شده است [۱]

بدین ترتیب به ازاء مقیاس‌های مختلف(r)، طول‌های مختلفی حاصل می‌گردد(P) که نمونه‌ای از آن در جدول ۲-۱ آورده شده بود. حال یک کاغذ لگاریتمی انتخاب کرده، بر روی محور xها مقیاس (r) و بر روی محور yها طول محاسبه شده با آن مقیاس (P) وارد می‌شود. از اتصال هر زوج r و P یک نقطه حاصل می‌شود. با بهم پیوستن نقاط بدست آمده، خطی با ضریب منفی(-d) به وجود می‌آید که رابط بین دو مقدار پیش گفته را بیان می‌کند(شکل ۲-۲).

از حاصل جمع عدد یک و ضریب d، مقداری حاصل می‌گردد که به آن بعد فراکتال گویند. لازم به ذکر است که در صورت مطالعه یک سطح، بعد فراکتال از افزودن عدد دو به ضریب d حاصل می‌گشت.



شکل ۲-۲ نحوه بدست آوردن شیب فراکتال خط ساحلی انگلستان[۱]

۲-۴ کاربرد بعد فراکتال

از بعد فراکتالی بویژه برای دو منظور می‌توان استفاده کرد. اول اینکه می‌توان هر شکل را با این کمیت معرفی کرد. به معنای دیگر می‌توان پیچ و تاب و تغییرپذیری آن را شناساند. بگونه‌ای که اشکال پرشکنج‌تر بعد فراکتال بزرگتر خواهند داشت. استفاده دوم عرصه‌ای است که نیاز به کاوش و بررسی بیشتر دارد. در این موارد می‌بایست ارتباط بین بعد فراکتال و ویژگی‌های پدیده مورد پژوهش را شناخت. به عنوان مثال نتیجه یک مطالعه بر روی سیستم گسل نهبندان ثابت کرده است در مناطقی که میدان اثر کرنش دو سطح وسیع گسترش یافته، از تراکم سطوح شکستگی کاسته شده و به عبارت دیگر سیستم گسله در آن مناطق جوانتر است [۴].

خوشبختانه علی‌رغم فعالیتهای زیادی که راجع به هندسه فراکتال صورت گرفته، همچنان امکان کار بر روی آن وجود دارد. در ادامه استفاده از هندسه فراکتال جهت شبیه‌سازی سطوح زمین‌شناسی شرح داده می‌شود.

۲-۵ کاربرد فراکتال در پردازش اطلاعات توپوگرافی

همانگونه که ذکر شد بسیاری از ساختارهای زمین‌شناسی، منجمله توپوگرافی زمین از قواعد آمار فراکتالی پیروی می‌کنند. دلیل این امر خصلت شکل گرفتن، فرسایش، جنس و شرایط محیطی حاکم بر یک منطقه است. به عنوان مثال آهک بدلیل سختی و خصلت شکست تیز احتمالاً مورفولوژی پرشکنجی خواهد داد و یا تبخیریها و یا رس‌ها بدلیل نرمی و قابلیت انحلال بالا،

مورفولوژی تپه‌ماهوری نمایش می‌دهند. البته بدیهی است که طول دوره‌ای که این سنگها در معرض فرسایش بوده‌اند نیز مهم است. به عبارت دیگر هر چه عمر رخنمون یک سنگ افزایش یابد، احتمالاً بدلیل فرسایش بیشتر سطح آن نرم‌تر خواهد بود و بالعکس. در ضمن شرایط محیطی نیز در شکل توپوگرافی نقش بازی می‌کند. بدین صورت که در منطقه‌ای با هوای گرم و مرطوب انتظار یک مورفولوژی نرم و در یک منطقه سرد و کوهستانی و پر برف انتظار یک مورفولوژی سخت و پر شکنج می‌رود.

حال انتظار اینست که در مورفولوژی‌های سخت و پرشکنج، بعد فراکتال سطح بزرگ و در مورفولوژی‌های نرم و آرام بعد فراکتال کوچک باشد. بدیهی است اگر روشی جهت محاسبه بعد فراکتال سطوح نقشه‌برداری شده وجود داشته باشد، بصورت کمی مورفولوژی را می‌توان معرفی نمود. سپس احتمالاً می‌توان تخمینگری را ابداع کرد که هنگام پردازش و افزایش چگالی آنها، بعد فراکتال سطح را تغییر ندهد. در این صورت می‌توان انتظار داشت که سطح شبیه‌سازی شده شباهت مناسبی با سطح حقیقی طبیعت داشته باشد.

برای بررسی موارد مذکور نیاز به یک سری اطلاعات دقیق نقشه‌برداری می‌باشد که در راستای اجرای این طرح نسبت به انجام برداشت‌های صحرایی نقشه‌برداری اقدام شده است. چگونگی برداشت اطلاعات و روش‌های پردازش در فصول آتی شرح داده خواهد شد.

۳

شرح عملیات نقشه‌برداری

۳-۱ مقدمه

منظور از برداشت، تعیین مختصات یا موقعیت نقاطیست از زمین که برای تهیه نقشه و یا شناسائی کلی مسطحاتی و ارتفاعی زمین و عوارض آن در یک منطقه لازمست [۵].

روش‌های مختلفی برای برداشت عوارض وجود دارد. به عنوان مثال می‌توان استفاده از دوربین‌های استادیمتری، تبدیل کننده به افق، پارالاکتیک، تبدیل کننده به افق، توتال[1] ، نیوو و یا استفاده از متر، ژالن، کمپاس و تخته سه پایه، بهره جستن از روش‌های فتوگرامتری و در نهایت به توسط GPS را نام برد.

بدیهی است هر روش مزایا و معایب خاص خود را دارد. به عنوان مثال روش استادیمتری سریع بوده اما از دقت خیلی بالایی برخوردار نیست. به عبارت دیگر یک وسیله مهندسی است. اما توتال هم سریع و هم دقیق است، اما این دوربین بسیار گران قیمت بوده و کمتر در اختیار است. روش‌های فتوگرامتری و یا استفاده از GPS دستی بسیار سریع هستند. اما دقت آنها بسیار پایین بوده و برای تهیه نقشه‌های کوچک مقیاس مناسب هستند.

به هر صورت در نقشه‌های مورد استفاده زمین‌شناسان و معدنچیان معمولاً مقیاس‌های متوسط و کوچک و با هزینه کم برداشت در نظر است. بنابراین در این طرح سعی شده است توسط دوربین RDS که نسبت به استادیمترها از دقت بیشتری برخوردار است، برداشت نقشه با مقیاس متوسط و توسط GPS برداشت مناطق با مقیاس کوچک صورت بگیرد تا کارکرد منطق فراکتال در پردازش داده‌های نقشه‌های با مقیاس متفاوت مورد مطالعه قرار بگیرد.

1- Total Station

۳-۲ برداشت توسط RDS

مناطقی از کردان و کرج جهت برداشت توسط دوربین RDS انتخاب گردیدند. ویژگی مورفولوژی منطقه کردان در کوهستانی بودن آن است. همچنین در این منطقه رخنمون‌هایی از رگه‌های سیلیس و باریت مشاهده می‌گردید که به صورت تیز و فرسایش نیافته بر شکنج منطقه افزوده است. هنگام برداشت سعی شده است که تمامی عوارض برداشت گردند، هر چند که بعید به نظر می‌رسد، شکست حاصل از وجود رگه‌های کم ضخامت باریت و سیلیس تأثیر خاصی در بعد فراکتال منطقه بگذارد. در این منطقه به توسط دو ایستگاه نقشه‌برداری حدود ۲۶ نقطه اطلاعاتی از محدوده‌ای به وسعت تقریبی ۱۰ هکتار برداشت شده است. بنابراین به طور متوسط از هر چهار صد متر مربع یک نقطه اطلاعاتی وجود دارد که مقیاس نقشه ۱:۲۰۰۰ خواهد بود. در ضمن سعی شده است عوارض با بلندی ۱/۵ متر برداشت شوند که در مقیاس نقشه خللی وارد نیاید. اطلاعات اولیه پردازش شده و به صورت فایل اطلاعات دکارتی جهت مطالعه بعد فراکتال و پردازش‌های دیگر آماده شده‌اند. این فایل اطلاعاتی در دیسک فشرده پیوست گزارش ارائه شده است.

منطقه دوم عملیاتی در مجاورت فرودگاه پیام، نرسیده به ماهدشت بوده است. مورفولوژی این منطقه تپه ماهوری است. از این منطقه ۲ نقشه تهیه شده است. برداشت یک مرتبه توسط RDS و مرتبه دیگر توسط GPS صورت گرفته است. بدیهی است در این صورت امکان مطالعه توانمندی روش‌های فراکتالی در تخمین بعد فراکتال نقشه‌های با مقیاس متفاوت فراهم خواهد گشت. ابتدا به نظر می‌رسد برداشت با GPS بدلیل دقت پایین، بعد فراکتال را نتواند به خوبی بیان نماید. البته این موضوع در فصول آتی مطالعه خواهد شد. به هر حال در برداشت با دوربین RDS از حدود ۲۰ هکتار حدود ۷۰۰ نقطه برداشت شده است. برای این منظور شش ایستگاه نقشه‌برداری درنظر گرفته شده و مقیاس نقشه حدوداً ۱:۱۵۰۰ خواهد بود.

۳-۳ برداشت توسط GPS

مناطقی از کرج و محلات جهت برداشت توسط GPS انتخاب گردیدند. همانگونه که سابقاً آورده شد، در منطقه کرج به این دلیل مجدداً با GPS برداشت صورت گرفت که اولاً منطقه تپه ماهور بوده است و ثانیاً بدینوسیله توانمندی عملکرد روش‌های تخمین بعد فراکتال و پردازش فراکتالی در موارد برداشت نقشه‌های کوچک مقیاس و بزرگ مقیاس، بررسی می‌شود. بنابراین منطقه مجاور فرودگاه پیام، نرسیده به ماهدشت توسط GPS نیز برداشت گردید. خصلت این داده‌ها چگالی کم آنها نسبت به اطلاعات برداشت شده توسط RDS است. در این مرحله حدود ۲۰۰ برداشت از محدوده حدود ۵۰ هکتار برداشت گردید. به عبارت دیگر حدوداً از هر ۲۵۰۰ متر مربع یک نقطه اطلاعاتی برداشت گردید. بنابراین مقیاس برداشت ۱:۵۰۰۰ خواهد بود. نکته‌ای که لازم به ذکر است اینکه GPS مورد استفاده از نوع گارمین ((Summit (etrex) انتخاب شده

است که از ویژگی‌های آن داشتن آلتی متر دیجیتال است. بنابراین هر چند که ممکن است خطای مطلق بیان ارتفاع توسط این وسیله به چند متر برسد، اما خطای نسبی ارتفاعی بین نقاط از حداکثر نیم متر تجاوز نمی‌کند (دقت آلتی متر). بنابراین واقعاً نقشه برداشت شده می‌تواند مقیاس ۱:۵۰۰۰ داشته باشد.

منطقه دوم عملیاتی توسط GPS در کیلومتر ۲۵ جاده محلات ـ خمین واقع شده است. از ویژگی‌های این منطقه می‌توان حضور هر دو مورفولوژی کوهستانی و ملایم را ذکر کرد. بدین ترتیب هر سه نوع مورفولوژی‌های کوهستانی، تپه ماهور و ملایم توسط GPS برداشت شده است. از طرف دیگر بعضاً شیب و شکست شیب در مناطق مطالعاتی محلات بقدری شدید می‌شد که امید می‌رود به توسط فراکتال تفاوت بعد سطوح مختلف قابل کشف و نمایش باشد. به هر صورت از این منطقه نیز حدود دو هزار نقطه در چند محدوده برداشت شده که مقیاس برداشت حدود ۱:۵۰۰۰ می‌باشد. تمامی اطلاعات برداشت شده در لوح فشرده پیوست گزارش ارائه شده است.

۳-۴ پردازش اطلاعات

تمامی اطلاعات برداشت شده وارد نرم افزار سورفر گردید. بر روی اطلاعات برداشت شده توسط RDS، پردازش نیز صورت گرفته و داده‌های خام تبدیل به مختصات دکارتی گردیدند. اطلاعات برداشت شده توسط GPS نیز که به صورت مختصات دکارتی بوده و نیازی به پردازش اولیه نخواهد داشت. سپس چارچوب نقشه‌ها مشخص شده، توسط تخمینگرهای مناسب در دسترس همچون کریجینگ، عکس مجذور فاصله و ... ، بر روی هر مجموعه داده‌ها فایل شبکه‌ای ساخته شده است. در نهایت نقشه‌توپوگرافی هر منطقه ترسیم شده است. در ادامه گزارش از این نقشه‌ها جهت قیاس نتایج حاصل از پردازش فراکتال و نتایج حاصل از روش‌های مرسوم بهره خواهیم جست.

۴

روشهای محاسبه بعد فراکتال

۴-۱ مقدمه

در این طرح، سه الگوریتم جهت محاسبه بعد فراکتال سطوح و خطوط ارائه شده و نرم‌افزارهای مربوطه ارائه گشته‌اند. هر کدام از الگوریتمها در شرایط خاصی جواب مناسب می‌دهند که در توضیح روش‌ها موارد مناسب استفاده از هر کدام نیز ذکر شده است. در زیر روشهای مذکور آورده شده‌اند.

۴-۲ روش پرگار تقسیم سیار

این روش اصولاً برای محاسبه بعد توپولوژی منحنی‌های دو بعدی استفاده می‌شود. روش کار به این صورت است که ابتدا طول منحنی مورد نظر با خط‌کشی به طول L اندازه‌گیری می‌شود. سپس در هر مرحله طول خط‌کش نصف می‌شود و سپس طول منحنی اندازه‌گیری می‌شود. نهایتاً نمودار لگاریتم طول منحنی در هر مرحله در برابر لگاریتم طول خط‌کش در هر مرحله رسم می‌گردد.

شیب خطی که به روش کمترین مربعات داده‌ها را درون‌یابی کند برابر با b خواهد بود که در حالت کلی b مقداری منفی است. در شکل ۴-۱ نموداری جهت نمایش لگاریتم طول خط‌کش در مقابل لگاریتم طول منحنی به عنوان نمونه آورده شده است. بعد فراکتال با استفاده از رابطه ۴-۱ محاسبه می‌شود.

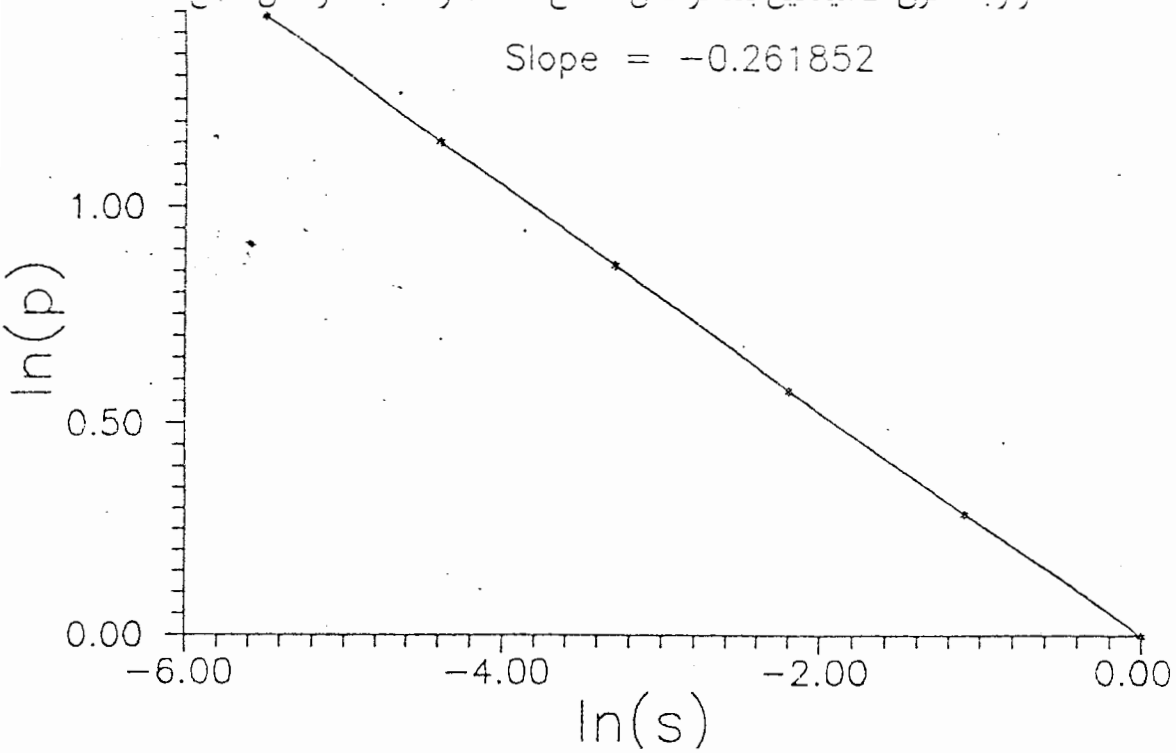
$$D = 1 - b \qquad (۱-۴)$$

البته این روش را با تغییراتی می‌توان برای محاسبه بعد فراکتال سطح نیز تعمیم داد. استفاده از این روش جهت محاسبه بعد فراکتال سطح به دو صورت انجام می‌شود که در زیر شرح داده می‌شوند.

الف- ابتدا یکسری مقاطع یک بعدی از سطح مورد نظر انتخاب می‌گردد. هر یک از مقاطع بعد فراکتال b-۱ دارند که از طریق روش پرگار تقسیم بدست آمده است. میانگین بعد فراکتال مقاطع مختلف را محاسبه کرده و بعد فراکتال سطح توسط رابطه ۴-۲ محاسبه می‌شود.

$$D' = 1 + D \qquad (۲-۴)$$

در رابطه فوق D میانگین بعد فراکتال مقاطع مختلف و ´D بعد فراکتال سطح است.



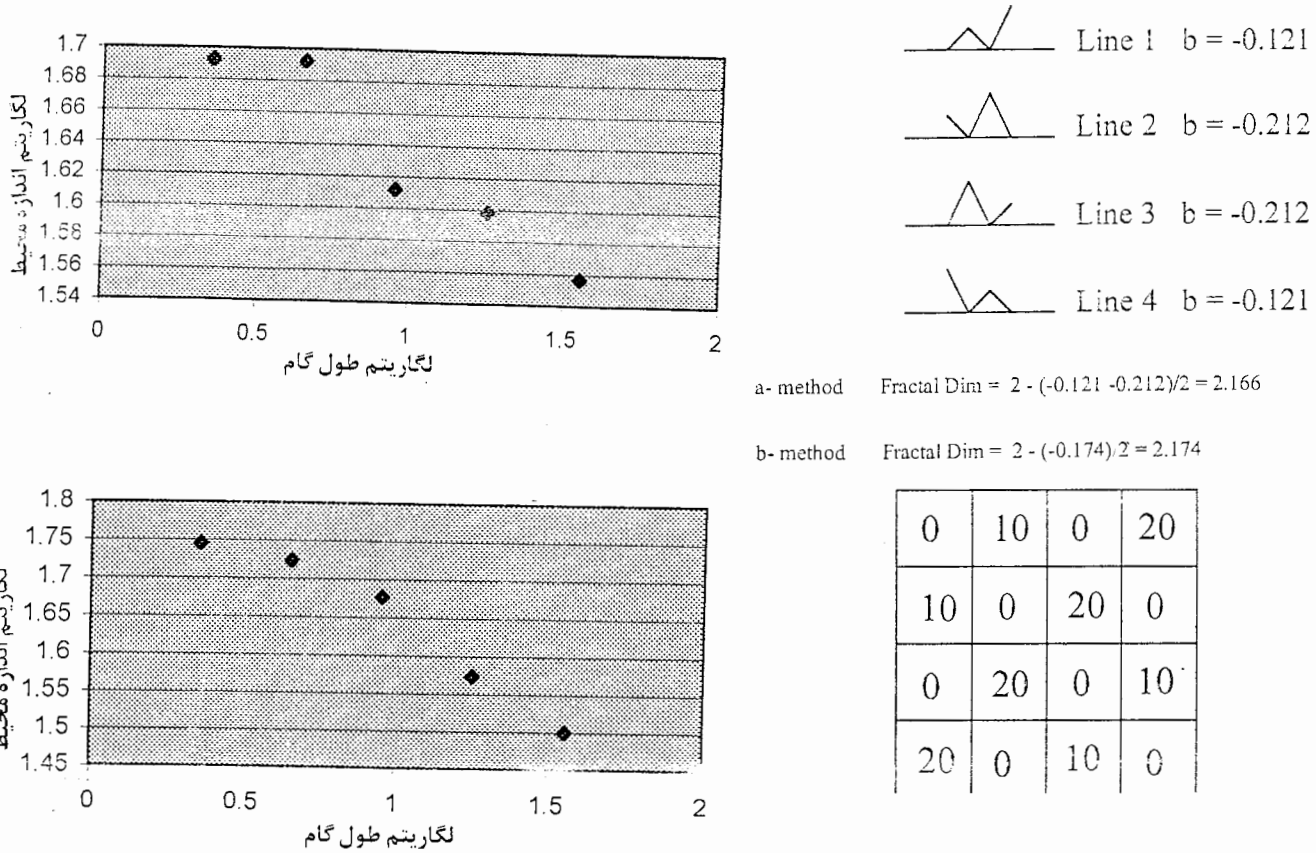
شکل ۴-۱ شکل شماتیک جهت نمایش لگاریتم طول خط کش در مقابل لگاریتم طول منحنی

ب- ابتدا با خط‌کش به طول L طول کلیه مقاطع به دست آمده و مجموع آنها محاسبه می‌گردد. در هر مرحله طول خط‌کش را نصف کرده و طول کلیه مقاطع با هم جمع زده می‌شوند. در نهایت نمودار لگاریتم جمع محیط مقاطع در مقابل لگاریتم طول خط‌کش رسم شده و بعد فراکتال سطح از فرمول ۴-۳ بدست می‌آید.

$$D = 2 - b \qquad (۳-۴)$$

در رابطه فوق D بعد فراکتال سطح و b شیب خط رگرسیون لگاریتم طول خط کش در مقابل لگاریتم محیط سطح است که عددی منفی است.

این روش برای اولین بار در این طرح بکار برده شده و به نظر می‌آید نتایج معتبرتری را نسبت به روش اول ارائه می‌کند. در شکل ۴-۲ نموداری جهت مقایسه نتایج حاصل از روش‌های *الف* و ب در محاسبه بعد فراکتال آورده شده است.



شکل ۴-۲ مقایسه روش‌های *الف* و ب پرگار تقسیم سیار در محاسبه بعد فراکتال

۴-۳ روش پرگار تقسیم سیار دو بعدی

ایـن روش که معادل دو بـعدی از روش پرگار تقسیم سیار یک بعدی می‌باشد توسط کلرک[1] (۱۹۸۶) ارائه گردیده است.

روش عمل به اینترتیب است که ابتدا سلول واحد شبکه را به عنوان گام اولیه در نظر می‌گیریم. مساحت این سلول برابر با حاصلضرب طول سلول شبکه در عرض سلول شبکه است. مقدار ارتفاع در چهار گوشه سلول مشخص است. مقدار میانگین این مقادیر به عنوان ارتفاع نقطه وسط سلول در نظر گرفته شده و از این نقطه به چهار گوشه سلول وصل می‌شود. مساحت رویه سلول برابر با جمع مساحت این چهار مثلث است. در نهایت مساحت کلی سطح برابر با جمع مساحت رویه‌ها خواهد بود.

1- Clerek

در مراحل بعدی طول و عرض سلول شبکه را دو برابر کرده و بدینترتیب مساحت سلول چهار برابر می‌شود. سپس مساحت کلی سطح محاسبه می‌شود.

نمودار لگاریتم مساحت کلی سطح در مقابل لگاریتم مساحت سلول شبکه رسم شده و خط رگرسیون منطبق بر داده‌ها ترسیم می‌شود. خط مذکور دارای شیب b می‌باشد و بعد فراکتال سطح از فرمول ۴-۴ بدست می‌آید [۶].

$$D = 2 - b \qquad (۴-۴)$$

۴-۴ استفاده از تغییر نما

بعد فراکتال با استفاده از تغییرنما توسط رابطه ۴-۵ محاسبه می‌گردد (باروق[1] (۱۹۸۱)) [۹].

$$D = (D_T + 1) - \frac{\lg \frac{\gamma(h_2)}{\gamma(h_1)}}{2\lg(\frac{h_2}{h_1})} \qquad (۵-۴)$$

در رابطه مذکور پارامترها عبارتند از:

D = بعد فراکتال

D_T = بعد اقلیدسی توپولوژی ساختار

h_2 و h_1 = فاصله جدایش

$\gamma(h)$ = مقدار تغییرنما برای فاصله جدایش h

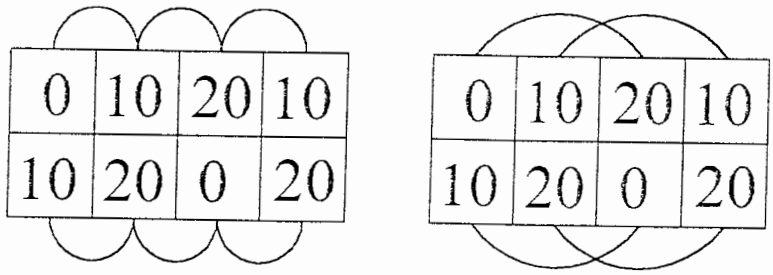
مقدار $\gamma(h)$ از فرمول ۴-۶ بدست می‌آید [۷]:

$$\frac{1}{2n}\sum_{i=1}^{n}(z_{x_{i+1}} - z_{x_i})^2 \qquad (۶-۴)$$

برای مثال در شکل ۴-۳، ۱= h_1 و ۲=h_2 انتخاب می‌شود. در اینصورت مقادیر تغییرنما به صورت زیر محاسبه خواهد شد.

$$h_1 = 1 \Rightarrow \gamma(h_1) = \frac{1}{12}\begin{pmatrix}(20-10)^2 + (0-20)^2 + (20-0)^2 \\ + (10-0)^2 + (20-10)^2 + (10-20)^2\end{pmatrix} = 62.5$$

1- Barough



شکل ۴-۳ یک شبکه نمونه جهت محاسبه مقادیر تغییرنما در گامهای مختلف

$$h_2 = 2 \Rightarrow \gamma(h_2) = \frac{1}{8}\left((0-10)^2 + (20-20)^2 + (20-0)^2 + (10-10)^2\right) = 100$$

بنابراین در شکل ۴-۳، بعد فراکتال به صورت زیر محاسبه می‌شود.

$$D = (2+1) - \frac{\log(\frac{100}{65.5})}{2\log(\frac{2}{1})} = 2.661$$

۴-۵ ارائه نرم‌افزار

بر اساس روش‌های فوق‌الذکر نرم‌افزار کامپیوتری بنام روش‌های فراکتال[1] تهیه گردید که می‌تواند بعد فراکتال سطوح مختلف را به هر یک از این روش‌ها محاسبه نماید. این نرم‌افزار توسط زبان برنامه‌نویسی دلفی[2] و برای سیستم عامل ویندوز طراحی گردیده است. نرم‌افزار FM قابلیت برقراری ارتباط با نرم‌افزار سورفر[3] تحت ویندوز از شرکت نرم افزار طلایی[4] را دارا می‌باشد. به این معنا که خروجی‌های نرم‌افزار سورفر به عنوان ورودی نرم‌افزار FM مورد استفاده قرار می‌گیرد و پس از انجام محاسبات مربوطه بعد فراکتال سطح را گزارش می‌گردد.

در ادامه به شرح نرم افزار و نحوه عملکرد کدهای برنامه پرداخته می‌شود.

۴-۵-۱ شرح عملکرد نرم‌افزار

فایل‌های خروجی نرم‌افزار سورفر توسط برنامه خوانده می‌شوند. لازم به ذکر است که نرم‌افزار سورفر برای شبکه‌بندی از سه فرمت خروجی استفاده می‌کند. فایل‌های با پسوند Grd. از نوع باینری

1- Fractal methods (FM)
2- Delfi
3- Surfer
4- Golden Saft Ware

یا اسکی و فایل‌های با پسوند Dat. از نوع Txt. می‌باشند. این فایل‌های با پسوند Dat. چون بصورت متنی هستند براحتی قابل دسترسی می‌باشند. سپس این فایل‌های متنی وارد نرم‌افزار FMمی‌شوند. نحوه کار به این صورت است که فایل متنی سطر به سطر خوانده شده و سپس اطلاعات آن در یک ماتریس مربعی منتقل می‌شود. نرم‌افزار FM به روش شئ گرا تهیه گردیده است. لذا جزئیات برنامه بر اساس نحوه ارتباط زیر روالها با کلاس مادر مورد بررسی قرار می‌گیرد. در شکل ۴-۴ شمای عملیاتی زیر روالهای برنامه ارائه گشته است.

۴-۵-۲ نحوه خواندن اطلاعات

ابتدا توسط رویه Read-data فایل ورودی داخل ماتریس دینامیکی داده‌ها قرا می‌گیرد. لازم به توضیح است به جهت استفاده بهینه از فضای حافظه موجود، کلیه ماتریس‌های برنامه به صورت پویا طراحی شده‌اند. به این معنا که در ابتدای شروع به کار برنامه طول آنها صفر می‌باشد. لکن در حین اجرای برنامه به هر مقدار که لازم باشد می‌توان به طول آنها افزود و یا از طولشان کاست. بدینترتیب این داده‌ها درون یک String grid قابل نمایش خواهند بود.

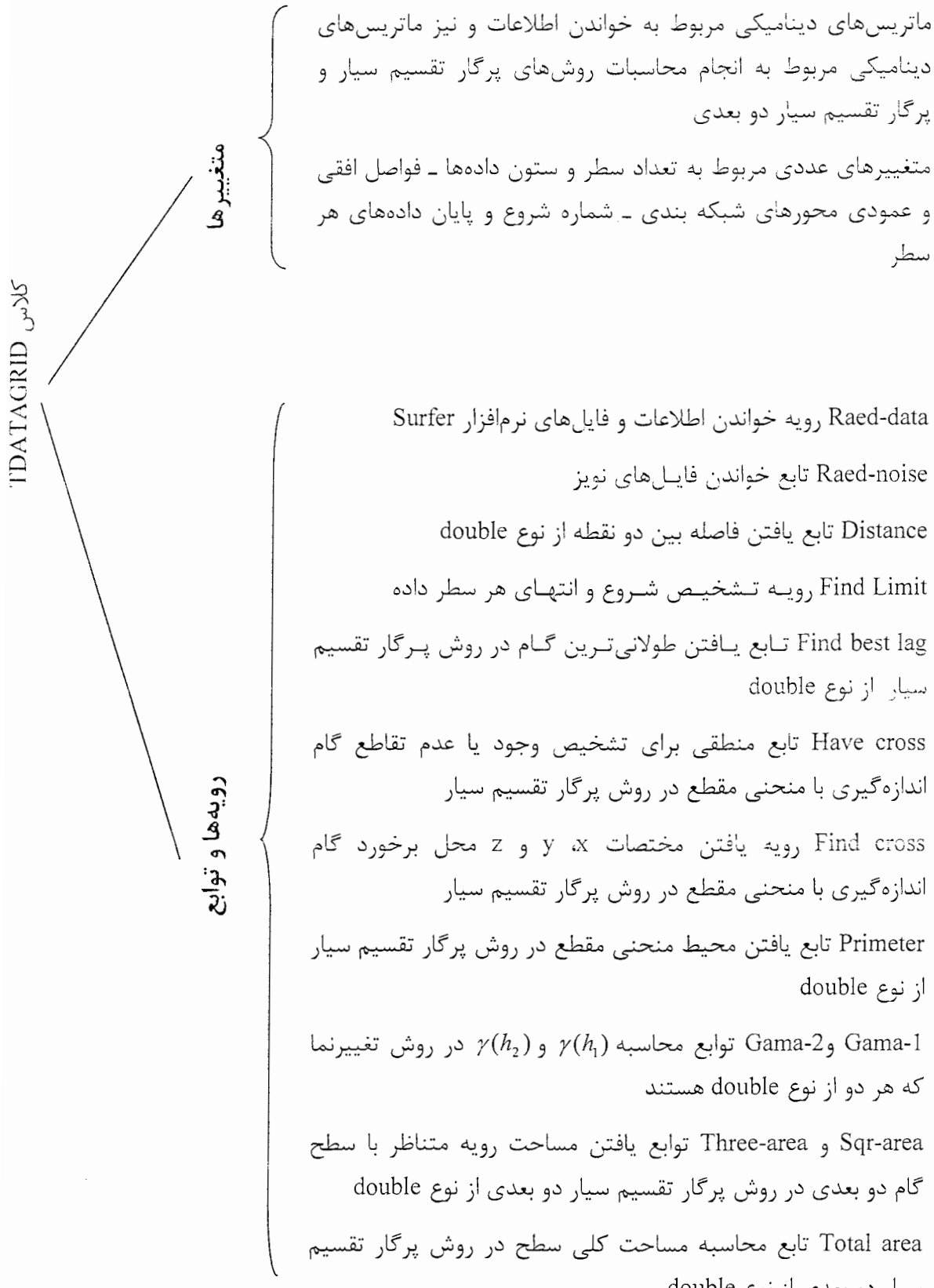
۴-۵-۳ محاسبه بعد فراکتال به روش پرگار تقسیم سیار

در این روش ابتدا تعداد مراحل نصف شدن گام (طول پرگار) وارد می‌گردد. سپس در یک ماتریس دو بعدی که دارای ۲ ستون می‌باشد، لگاریتم گام و محیط مربوط به هر گام محاسبه می‌گردد. تعداد سطرهای این ماتریس برابر تعداد مراحل نصف شدن گام می‌باشد. در جدول ۴-۱ شمای ماتریس ذخیره نتایج آورده شده است. در شکل ۴-۵ جزئیات عملکرد فرمت ب روش پرگار تقسیم سیار ارائه شده است.

جدول ۴-۱ شمای ماتریس ذخیره نتایج در روش پرگار تقسیم سیار

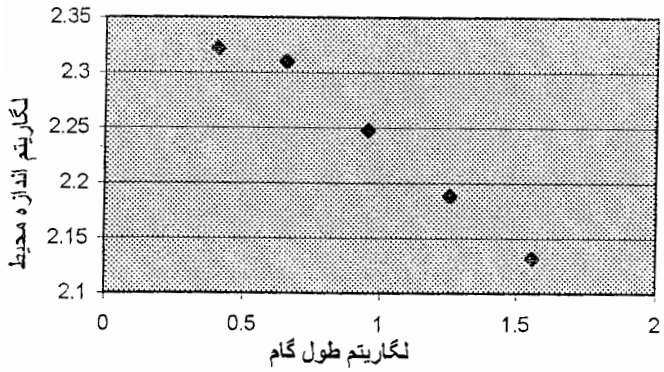
گام	محیط
L	P_1
$L/2$	P_2
$L/4$	P_3
$L/2^{n-1}$	P_n

n = تعداد مراحل نصف شدن گام



شکل ۴-۴ شمای عملیاتی زیرروالهای برنامه FM

		مسیر اول	مسیر دوم	مسیر سوم	مسیر چهارم
گام اول	طول گام	۳۶,۰۵۵	۳۶,۰۵۵۵	۳۶,۰۵۵۵	۳۶,۰۵۵۵
	محیط مسیر	۳۶.۰۵۵	۳۱,۶۲۲۸	۳۱,۶۲۲۸	۳۶,۰۵۵۵
	تعداد تکرار	۱	۰,۸۷۷	۰,۸۷۷	۱
گام دوم	طول گام	۱۸,۰۲۷۷	۱۸,۰۲۷۷	۱۸,۰۲۷۷	۱۸,۰۲۷۷
	محیط مسیر	۳۹,۶۷۰۸	۳۷,۴۹۷۷	۳۷,۴۹۷۷	۳۹,۶۷۰۸
	تعداد تکرار	۲,۲۰۰	۲,۰۷	۲,۰۷	۲,۲۰۰
گام سوم	طول گام	۹,۰۱۳۸	۹,۰۱۳۸	۹,۰۱۳۸	۹,۰۱۳۸
	محیط مسیر	۴۰,۹۱۰۷	۴۷,۵۴۹۶	۴۷,۵۴۹۶	۴۰,۹۱۰۷
	تعداد تکرار	۴,۵۳	۵,۲۷۵	۵,۲۷۵	۴,۵۳
گام چهارم	طول گام	۴,۵۰۶۹	۴,۵۰۶۹	۴,۵۰۶۹	۴,۵۰۶۹
	محیط مسیر	۴۹,۲۳۶	۵۲,۹۲۳۳	۵۲,۹۲۳۳	۴۹,۲۳۶
	تعداد تکرار	۱۰,۹۲۴۶	۱۱,۷۴۲۶	۱۱,۷۴۲۶	۹۲۴۶.۱۰
گام بنجم	طول گام	۲.۵۳۳۵	۲.۵۳۳۵	۲,۵۳۳۵	۲,۵۳۳۵
	محیط مسیر	۴۹,۲۸۰۸	۵۵,۶۲۱۸	۵۵,۶۲۱۸	۴۹,۲۸۰۸
	تعداد تکرار	۲۱,۸۶۸۹	۲۴,۶۸۲۹	۲۴,۶۸۲۹	۲۱,۸۶۸۹



0	10	0	20
10	0	20	0
0	20	0	10
20	0	10	0

Fractal Dim = 2 - (-0.174)/2 = 2.174

شکل ۴-۵ جزئیات عملکرد فرمت ب روش پرگار تقسیم سیار

بعد فراکتال بر طبق الگوریتم زیر محاسبه خواهد شد.

الف- در هر سطح ماتریس داده، توسط رویه Find-limit محدوده شروع و پایان داده معتبر مشخص می‌گردد.

تبصره: داده معتبر داده‌ای است که شامل Z (میزان ارتفاع) یک نقطه می‌باشد و در واقع داده‌ای است که توسط نرم‌افزار سورفر بلانک[1] شده است.

ب- در هر سطح طولانی‌ترین گام ممکن برای محاسبه محیط عبارت از فاصله بین دو سر ابتدا و انتهای داده‌های معتبر سطر است. لذا این گام به عنوان طولانی‌ترین گام در نظر گرفته می‌شود.

ج- مراحل الف و ب برای کلیه سطرهای ماتریس داده انجام می‌شود و بزرگترین مقدار گام بدست آمده، توسط رویه Find-best-lag بعنوان گام اصلی شروع محاسبه محیط در نظر گرفته می‌شود.

د- تعداد مراحل نصف شدن گام توسط نرم‌افزار از کاربر استعلام و معین می‌گردد.

هـ- به ازاء مقدار گام، محیط کلیه سطرها محاسبه شده و با هم جمع می‌گردد.

و- مقدار گام و محیط مربوطه در ماتریس مربوط به ذخیره نتایج ثبت می‌گردد.

ز- گام نصف می‌شود.

ح- مراحل هـ و و ز به تعداد دفعاتی که در مرحله د مشخص شده انجام می‌گیرد.

ط- ارسال نتایج محاسبات و مقادیر ماتریس ذخیره نتایج به نمودار گرافیکی و رسم نمودار لگاریتم محیط در برابر لگاریتم گام اندازه‌گیری صورت می‌گیرد.

در اینصورت بعد فراکتال از رابطه ۴-۷ بدست می‌آید.

$$D = 1 + (1 - b) \qquad (۴-۷)$$

در رابطه فوق b شیب نمودار است.

۴-۵-۴ محاسبه بعد فراکتال به روش تغییر نما

پس از خواندن اطلاعات ورودی و ارسال اطلاعات به ماتریس داده مراحل زیر انجام می‌گیرد.

الف- تعیین فواصل جدایش h_1 و h_2 برای محاسبه تغییرنما

1- Blank

ب- در هر سطر ماتریس داده، توسط رویه Find-limit محدوده داده معتبر مشخص می‌گردد.

ج- به ازاء فاصله جدایش h_2 مقدار تابع Gama-2 برای کلیه سطرهای ماتریس داده محاسبه می‌گردد.

د- به ازاء فاصله جدایش h_1 مقدار تابع Gama-1 برای کلیه سطرهای ماتریس داده، محاسبه می‌گردد.

هـ- مقدار تغییرنما و سپس بعد فراکتال از رابطه ۴-۵ گزارش می‌گردد.

۴-۴-۵ محاسبه بعد فراکتال به روش پرگار تقسیم سیار دو بعدی

در این روش ابتدا تعداد مراحل دو برابر شدن طول گام اولیه و یا چهار برابر شدن مساحت گام اولیه مشخص می‌گردد.

سپس در یک ماتریس دو بعدی که دارای دو ستون می‌باشد، لگاریتم مساحت گام دو برابر لگاریتم مساحت سطح کلی ثبت می‌گردد. تعداد سطرهای این ماتریس برابر تعداد مراحل چهار برابر شدن سطح گام می‌باشد. در جدول ۴-۲ شمای ماتریس ذخیره نتایج آورده شده است.

جدول ۴-۲ ماتریس ذخیره نتایج در روش پرگار تقسیم سیار دو بعدی

مساحت گام	مساحت کلی
S_1	A_1
4S	A_2
16S	A_3
$4^{n-1}S$	A_n

n= تعداد مراحل ۴ برابر شدن سطح

نحوه محاسبه بعد فراکتال بر طبق الگوریتم زیر خواهد بود.

الف- کوچکترین گام که در واقع همان مساحت سلول واحد شبکه می‌باشد به عنوان گام اولیه در نظر گرفته می‌شود.

ب- تعداد مراحل چهار برابر شدن سطح گام از کاربر پرسیده می‌شود.

ج- توسط تابع Three-area سطح پوسته هرسلول شبکه به چهار مثلث تقسیم می‌گردد. ارتفاع (Z) نقطه وسط هر سلول از میانگین‌گیری ارتفاع در گوشه‌ها بدست می‌آید. مساحت این چهار مثلث محاسبه می‌گردد.

د- توسط تابع Total-area مساحت کلی سطح با استفاده از نتایج مرحله ج بدست می‌آید.

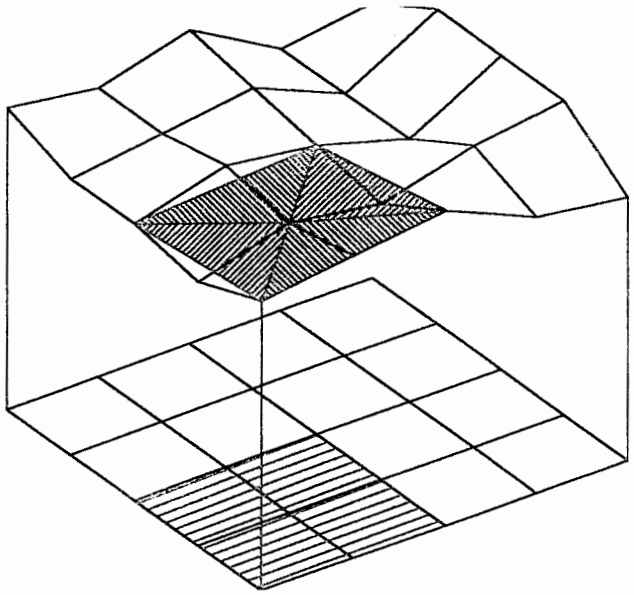
هـ- مقدار مساحت گام و مساحت کلی سطح متناظر با آن در ماتریس نتایج ثبت می‌گردد.

و- سطح گام چهار برابر می‌شود.

ز- مراحل ج، د، هـ و و به تعداد دفعات تعیین شده در مرحله ب تکرار می‌شود.

ح- ارسال نتایج محاسبات و مقادیر ماتریس ذخیره نتایج به نمودار گرافیکی و رسم نمودار لگاریتم مساحت کلی سطح در برابر لگاریتم مساحت گام اندازه‌گیری صورت می‌گیرد. بدینترتیب شیب نمودار (b) محاسبه می‌گردد. بعد فراکتال با استفاده از رابطه ۴-۴ بدست می‌آید. جهت روشن شدن موضوع در شکل ۴-۶ مفاهیم مساحت گام و مساحت کلی سطح نمایش داده شده است.

لازم به تذکر است که این نرم‌افزار توانایی محاسبه بعد فراکتال سطوح بلانک شده در نرم‌افزار سورفر را به روش پرگار تقسیم سیار دو بعدی ندارد. در ضمن این برنامه قادر است تا به کمک تابع Read-noise فایل‌های مربوط به نویزهای تولید شده در نرم‌افزار دیگری بنام شبیه‌سازی فراکتال[1] را خوانده و به کمک روش‌های پرگار تقسیم سیار و تغییرنما بعد فراکتال آن را محاسبه نماید.



شکل ۴-۶ نمایشی از مفاهیم مساحت کلی سطح و مساحت گام در روش پرگار تقسیم سیار دو بعدی

1- Fractal Simulation (FM)

۵

تولید داده‌های معتبر آماری با بعد فراکتال معین

۵-۱ مقدمه

در این فصل روشهایی جهت تولید اعداد براونی ارائه گردیده و همچنین تخمینگری جهت افزایش چگالی داده‌ها بگونه‌ایکه در مراحل شبیه‌سازی سطوح بعد فراکتال تغییر نکند شرح داده خواهد شد.

۵-۲ روشهای تولید نویز (آشفتگی) براونی[1]

مبانی تولید حرکت براونی، بر اساس حرکت تصادفی ذرات استوار است. بدین معنا که یک ذره به ازای هر واحد افزایش در محور x می‌تواند یک یا چند واحد در راستای محور y افزایش یا کاهش داشته باشد [۸]. از این روش می‌توان برای تولید اعداد تصادفی براونی استفاده نمود.

از نقطه نظر زمین آماری، حرکت براونی، مدل مفید و مهمی جهت مطالعه آماری ساختارهای زمین شناسی است. برای تولید حرکت براونی نیاز به یکسری ورودی‌هایی از اعداد تصادفی توزیع طبیعی با میانگین صفر و واریانس واحد داریم که اصطلاحاً اعداد تصادفی گوسی نامیده می‌شوند. روش‌های تولید این گونه اعداد تصادفی بسیار توسعه یافته‌اند و در این مبحث نیازی به شرح چگونگی تولید آنها نمی‌باشد.

1- Broun noise

در این نرم‌افزار به دو روش می‌توان حرکت براونی را شبیه‌سازی کرد. اول به کمک اعداد تصادفی گوسی و الگوریتم تولید حرکت براونی کسری و دوم با استفاده از الگوریتم جابجایی نقطه میانی می‌توان حرکت براونی را تولید کرد.

ابتدا هر یک از روش‌های فوق‌الذکر شرح داده می‌شوند.

۵-۳ الگوریتم تولید حرکت براونی کسری یک بعدی

به دو روش از این الگوریتم جهت تولید حرکت براونی کسری یک بعدی استفاده شده است. در زیر این دو روش شرح داده می‌شوند.

۵-۳-۱ روش اول تولید حرکت براونی کسری یک بعدی

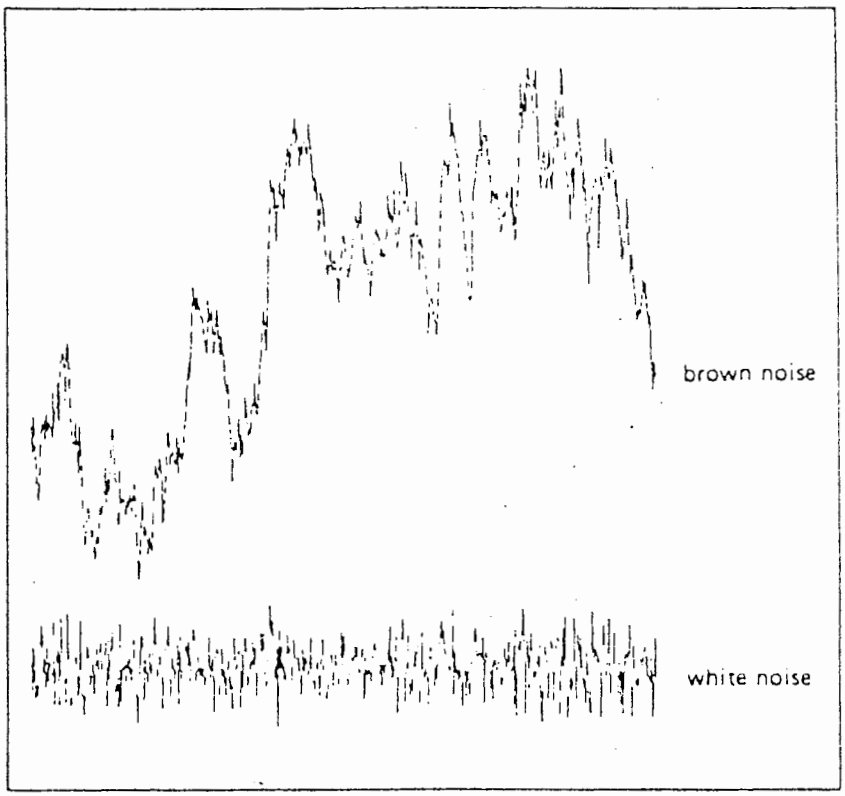
در این روش یک مقدار اولیه به عنوان نقطه شروع حرکت براونی تولید می‌شود. سپس با استفاده از رابطه ۵-۱ عدد تصادفی بعدی تولید می‌شود [۶].

$$x_{(T_2)} - x_{(T_1)} \alpha (T_2 - T_1)^H \qquad (۱-۴)$$

چنانچه $H = 1/2$ باشد حرکت براونی تصادفی استاندارد تولید خواهد شد که بعد فراکتال آن نیز ۱/۵ است. اما چنانچه H هر عدد دیگری در فاصله صفر و یک باشد (0<H<1) حرکت براونی کسری با بعد فراکتال D=2-H بدست می‌آید. در شکل ۵-۱ پارازیت‌های سفید و همچنین پارازیت‌های براونی مربوطه به عنوان نمونه آورده شده‌اند. همچنین در شکل ۵-۲ مثالی از تولید اعداد تصادفی حرکت براونی کسری برای دو مقدار متفاوت H آورده شده است.

۵-۳-۲ روش دوم تولید حرکت براونی کسری یک بعدی

این روش به الگوریتم جابجایی نقطه میانی موسوم است. این الگوریتم به دقت از مفهوم تکرار جابجاسازی (همچون منحنی سه‌گوش کوچ) پیروی می‌کند [۶]. در هر مرحله، از نتایج مرحله قبل به عنوان ورودی استفاده می‌شود. نقطه میانی قطعه اول به صورت متوسط دو نقطه انتهایی به علاوه یک جبرانی تصادفی محاسبه می‌گردد. اندازه جبرانی تصادفی متناسب با اندازه قطعه است. این فرآیند روی دو قطعه جدیدی که شکل گرفته تکرار می‌شود تا شکل نهایی حاصل بشود [۶].

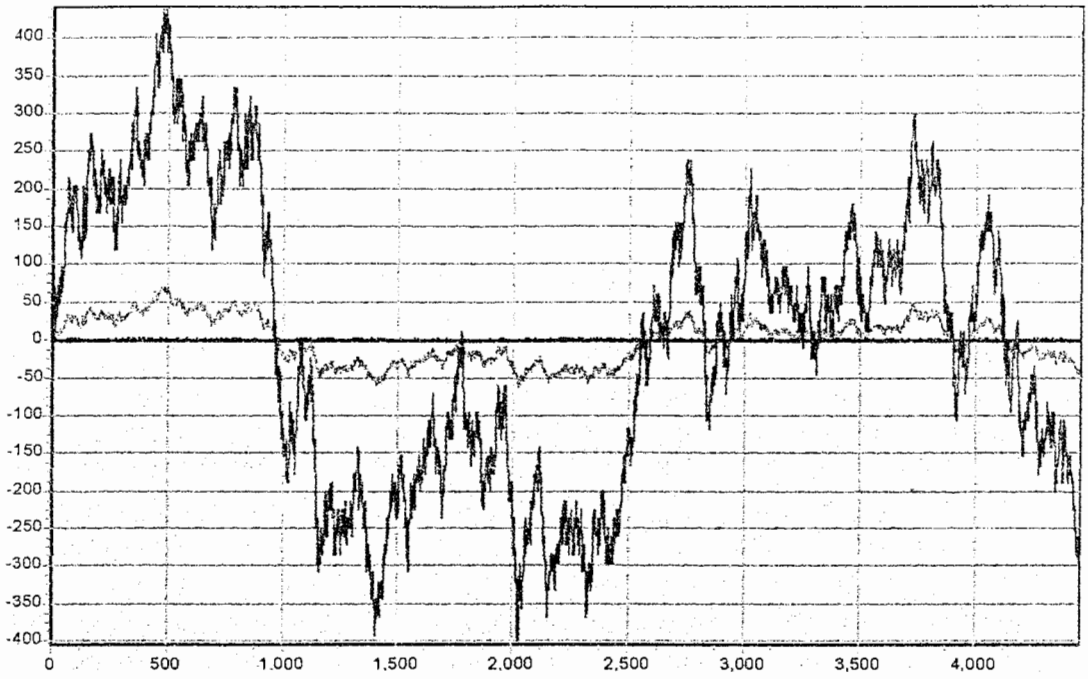


شکل ۵-۱ حرکت براونی با پارازیت‌های سفید آن (پیتجن و سوپ ۱۹۸۸) [۶]

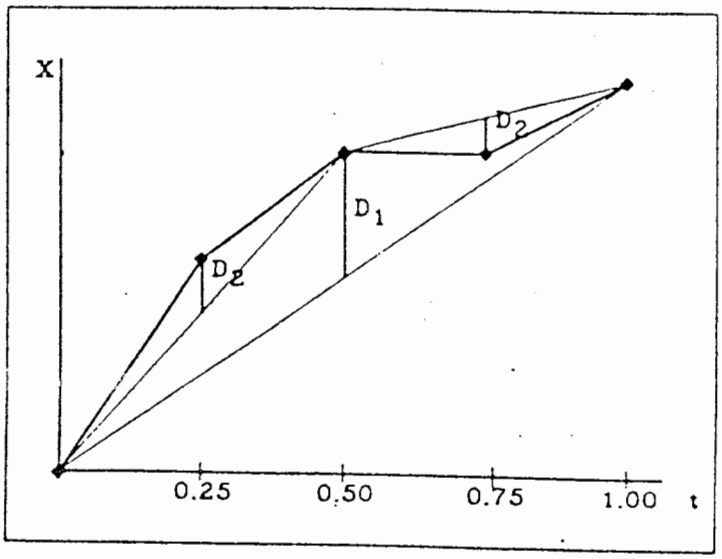
در شکل ۵-۳ به صورت شماتیک مثالی از جابجایی نقطه میانی برای حرکت براونی یک بعدی آورده شده است. لازم به ذکر است که اندازه جبرانی تصادفی، ضریبی متناسب با طول قطعه از یک عدد تصادفی گوسی است.

۵-۴ حرکت براونی کسری دو بعدی

الگوریتم حرکت براونی کسری دو بعدی را می‌توان بر پایه تکنیک جابجایی نقطه میانی یک بعدی توسعه داد. نقطه میانی یک سلول شبکه مربعی به صورت متوسط چهار نقطه گوشه‌ای بعلاوه یک جبرانی تصادفی متناسب با فاصله نقطه میانی از نقاط گوشه‌ای محاسبه می‌گردد. همچنین از این تکنیک می‌توان بر روی شبکه سه ضلعی متساوی الاضلاع استفاده نمود. ارتفاع نقطه میانی سه ضلعی برابر است با متوسط ارتفاع سه نقطه گوشه‌ای بعلاوه جبرانی تصادفی که متناسب با میانگین فواصل است[۶].



شکل ۵-۲ مثالی از تولید اعداد تصادفی حرکت براونی کسری برای دو مقدار متفاوت از H که توسط نرم افزار تهیه شده، رسم شده است



شکل ۵-۳ جابجایی نقطه میانی برای حرکت براونی یک بعدی

در اینجا مختصراً الگوریتم حرکت براونی کسری روی شبکه مربعی توضیح داده می‌شود.

در هر مرحله از فرآیند ترکیب، نقطه میانی سلول شبکه مربعی از مرحله قبل حاصل می‌شود. اگر اندازه ضلع شبکه معمولی S باشد، شبکه جدید که ۴۵ درجه نسبت به شبکه معمولی می‌چرخد، اندازه سلولی برابر با $\frac{S}{\sqrt{2}}$ خواهد داشت. در هر تکرار اندازه سلول شبکه در ضریب $r = \frac{1}{\sqrt{2}}$ ضرب خواهد شد [۶]. در شکل ۵-۴ سه مرحله اول الگوریتم جابجایی نقطه میانی بر روی یک شبکه منظم نشان داده شده است.

بر اساس روش‌های فوق‌الذکر نرم‌افزار کامپیوتری تهیه گردید که می‌تواند حرکت براونی را تولید کند. هدف اصلی که در تهیه این نرم‌افزار دنبال شده، ایجاد روشی برای تولید حرکت براونی با استفاده از الگوریتم جابجایی نقطه براونی، بدون تغییر تغییرنمای داده‌های اولیه (بعد فراکتال سطح اولیه) است.

این نرم‌افزار توانایی استفاده از خروجی‌های شبکه‌بندی شده نرم‌افزار سورفر با فرمت dat. را دارد. همچنین می‌تواند داده‌های بلانک شده را نیز به عنوان ورودی بپذیرد.

فایل‌های با پسوند dat. که به صورت Space delimiter جدا شده و نحوه ساخت آنها در فصل بعد توضیح داده خواهد شد، به عنوان ورودی این نرم‌افزار هستند.

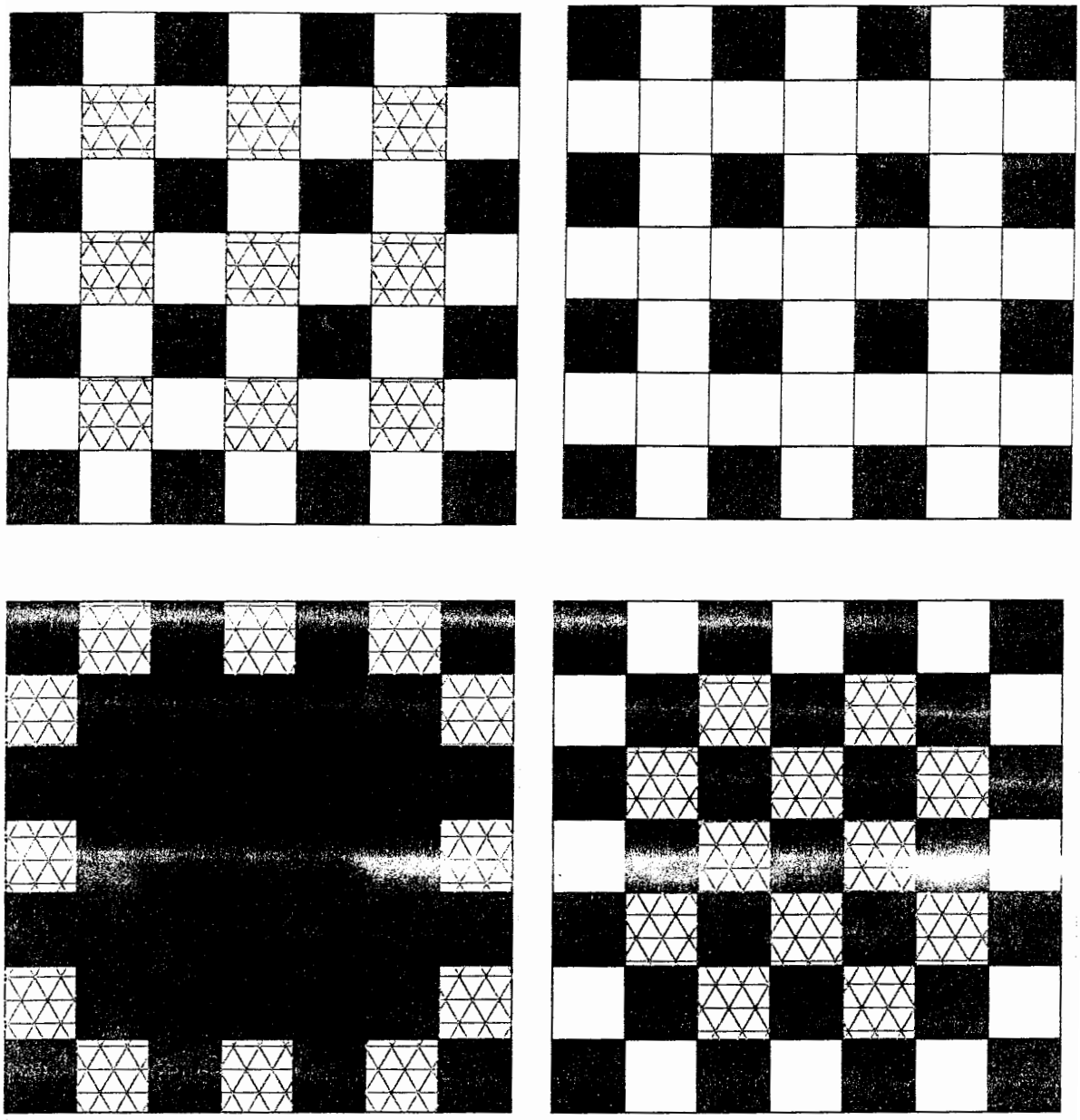
فایل متنی توسط نرم‌افزار FM سطر خوانده شده و در یک ماتریس مربعی قرار داده می‌شود. در نهایت محاسبات بر روی آن انجام می‌شود. سپس تغییرنمای پایه این داده‌ها محاسبه می‌گردد و اعداد تصادفی گوسی نیز تولید می‌شوند. نهایتاً توسط الگوریتم جابجایی نقطه میانی ضریبی متناسب با طول شبکه این اعداد گوسی برای ایجاد مدل جابجایی نقطه میانی بکار می‌رود.

این ضریب باید بگونه‌ای توسط کاربر انتخاب شود که تغییرنمای اولیه تغییر نکند. در شکل ۵-۵ جزئیات برنامه بر اساس ارتباط زیر روال‌ها با کلاس مادر شرح داده شده است.

۵- ۵ نحوه خواندن اطلاعات

ابتدا توسط رویه Read-Data فایل ورودی درون ماتریس داده‌ها قرار می‌گیرد. سپس این داده‌ها درون یک String Grid نمایش داده می‌شوند.

String Grid به تعداد سطر و ستون‌های شبکه داده‌ها می‌باشد. در ردیف صفر اعداد متعلق به محور x های داده‌ها نمایش داده می‌شود که در واقع تفاضل هر دو عدد مجاور برابر مقدار x Spacing شبکه داده‌ها است. در ستون صفر هم اعداد متعلق به محور y های داده‌ها نمایش داده می‌شود که تفاضل هر دو عدد مجاور برابر مقدار y Spacing شبکه داده‌ها می‌باشد.



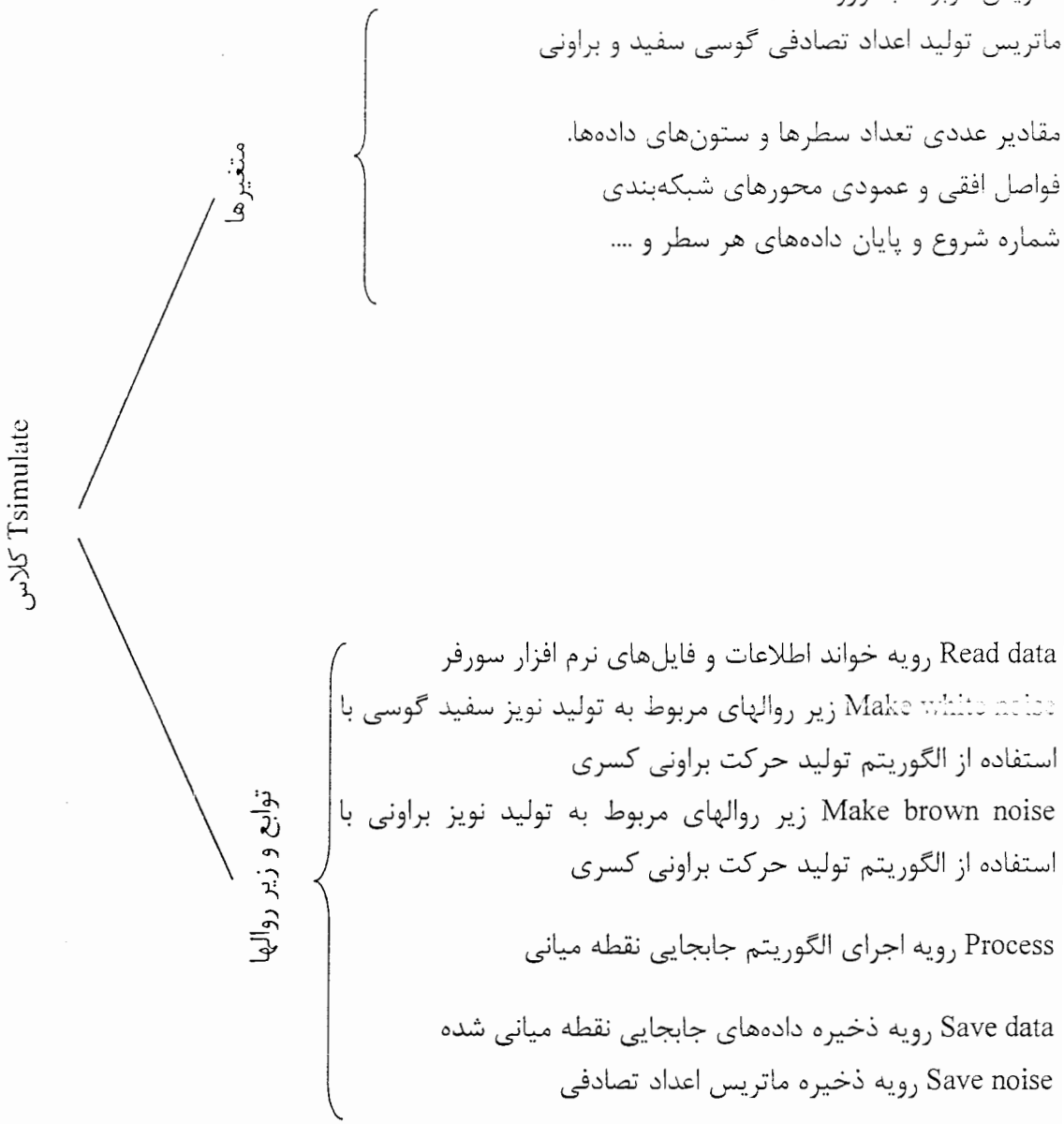
داده‌های مربوط به شبکه اولیه

داده‌های مربوط به اجرای مرحله اول الگوریتم جابجایی نقطه میانی

داده‌های مربوط به اجرای مرحله دوم الگوریتم جابجایی نقطه میانی

داده‌های مربوط به اجرای الگوریتم جابجایی نقطه میانی بر روی نقاط مرزی

شکل ۵-۴ سه مرحله اول الگوریتم جابجایی نقطه میانی بر روی یک شبکه منظم



شکل ۵-۵ شمای عملیاتی زیر روالهای برنامه شبیه‌ساز فراکتال (FS) و ارتباط زیر روالها با کلاس مادر

۵-۶ ساختن ماتریس اعداد تصادفی گوسی

چنانچه شبکه ورودی دارای x ستون و y ردیف باشد، می‌بایست پس از یک مرحله انجام جابجایی نقطه میانی به یک شبکه با (2x-1) ستون و (2y-1) سطر تبدیل شود. لذا می‌بایست تعداد $[(2x-1)(2y-1)-(xy)]$ عدد تصادفی گوسی تولید شود. این عمل توسط رویه Make-white-noise صورت می‌گیرد و اعداد تولید شده در یک ماتریس خطی دینامیکی قرار داده می‌شوند.

۵-۷ ساختن ماتریس اعداد نویز براونی توسط الگوریتم تولید حرکت براونی کسری

با استفاده از اعداد تصادفی تولید شده در مرحله قبل، در این مرحله اعداد تصادفی Brovn noise تولید می‌گردند. این عمل توسط رویه Make-white-noise صورت می‌گیرد. تعداد اعداد تولید شده نویز براونی به همان تعداد اعداد گوسی می‌باشد. این اعداد نیز درون یک ماتریس خطی دینامیکی قرار می‌گیرند.

۵-۸ محاسبه بعد فراکتال به روش تغییرنما

در این مرحله بعد فراکتال داده‌های ورودی همانند روشی که در نرم‌افزار FM شرح داده شد محاسبه می‌گردد و عدد محاسبه شده توسط یک Message box به کاربر ارائه می‌گردد.

۵-۹ اجرای الگوریتم جابجایی نقطه میانی

این عمل توسط رویه Process صورت می‌پذیرد.

در ذیل این رویه شرح داده می‌شود.

داده‌هایی که قرار است توسط الگوریتم جابجایی نقطه میانی تخمین زده شوند به سه گروه تقسیم می‌گردند.

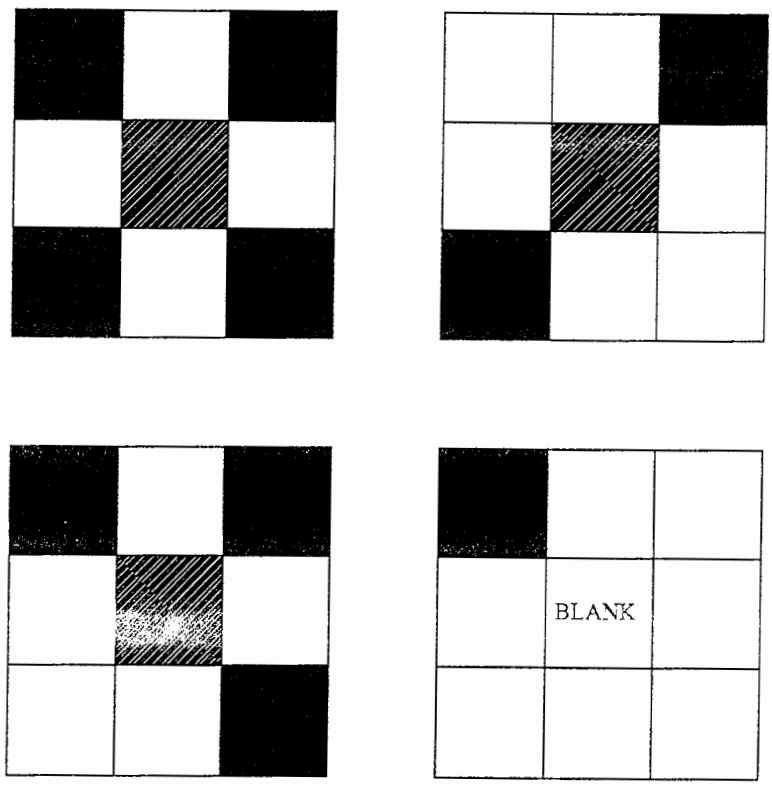
الف- داده‌هایی که برای اجرای الگوریتم جابجایی نقطه میانی طول شبکه‌ای برابر طول شبکه اولیه داده‌ها برایشان در نظر گرفته می‌شود.

ب- داده‌هایی که طول شبکه‌شان در ضریب $R = \frac{1}{\sqrt{2}}$ ضرب می‌شود. این داده‌ها متعلق به مرحله دوم جابجایی نقطه میانی هستند و نسبت به شبکه اولیه 45° چرخش دارند.

ج- داده‌هایی که در لبه مرزی شبکه و یا مرز منطقه بلانک شده قرار دارند. این داده‌ها نیز بسته به اینکه متعلق به کدام مرحله جابجایی باشند (مرحله اول یا دوم) خود به ۲ گروه تقسیم می‌شوند:

اول داده‌هایی که فقط یک همسایه معتبر دارند و دوم داده‌هایی که بیش از یک همسایه معتبر دارند.

داده‌های گروه اول بلانک در نظر گرفته می‌شوند. لکن داده‌های گروه دوم مشابه داده‌های مراحل الف و ب محاسبه می‌گردند. در شکل ۵-۶ انواع حالات ممکن به لحاظ مجاورت داده معتبر با داده‌های بلانک شده و رفتار نرم افزار با هر کدام از این حالات نمایش داده شده است.



شکل ۵-۶ انواع حالات ممکن به لحاظ مجاورت داده معتبر با داده‌های بلانک شده و رفتار نرم افزار با هر کدام از این حالات

ارتفاع هر داده تخمین زده شد برابر میانگین ارتفاع چهار نقطه یا کمتر از همسایگانش در سلول شبکه، بعلاوه یک جبرانی تصادفی که ضریبی متناسب با طول شبکه از اعداد تصادفی گوسی است، در نظر گرفته می‌شود.

این ضریب در ابتدای شروع رویه Process توسط کاربر تعیین می‌شود.

نهایتاً بعد فراکتال شبکه چگال شده جدید توسط روش تغییرنما گزارش می‌گردد. رویه Process آنقدر انجام می‌پذیرد تا با بدست آوردن ضریب مناسب بتوان شبکه اولیه را بدون تغییر بعد فراکتال چگال نمود.

۵-۱۰ نحوه ذخیره محاسبات

برای این کار دو رویه Save data و Save noise در نظر گرفته شده است.

رویه Save data شبکه نهایی را درون فایل با فرمت dat. و از نوع متنی (text) و بصورت (Tab delimited) ذخیره می‌نماید که این فایل هم توسط نرم‌افزار سورفر و هم نرم‌افزار FM و هم به عنوان ورودی اولیه برای ایجاد یک شبکه چگال‌تر در نرم‌افزار شبیه‌ساز فراکتال قرار می‌گیرند.

این فایل‌ها توسط نرم‌افزار FM قابل بازخوانی برای محاسبه بعد فراکتال‌شان هستند. همچنین می‌توانند توسط نرم‌افزارهای مختلف ترسیم نمودار مانند Exel و ... نیز بازخوانی شوند.

۶

نحوه کار با نرم‌افزارهای FM و FS

۶-۱ مقدمه

در این فصل به تفصیل راجع به عملکرد نرم‌افزارهای نوشته شده توضیح داده می‌شود.

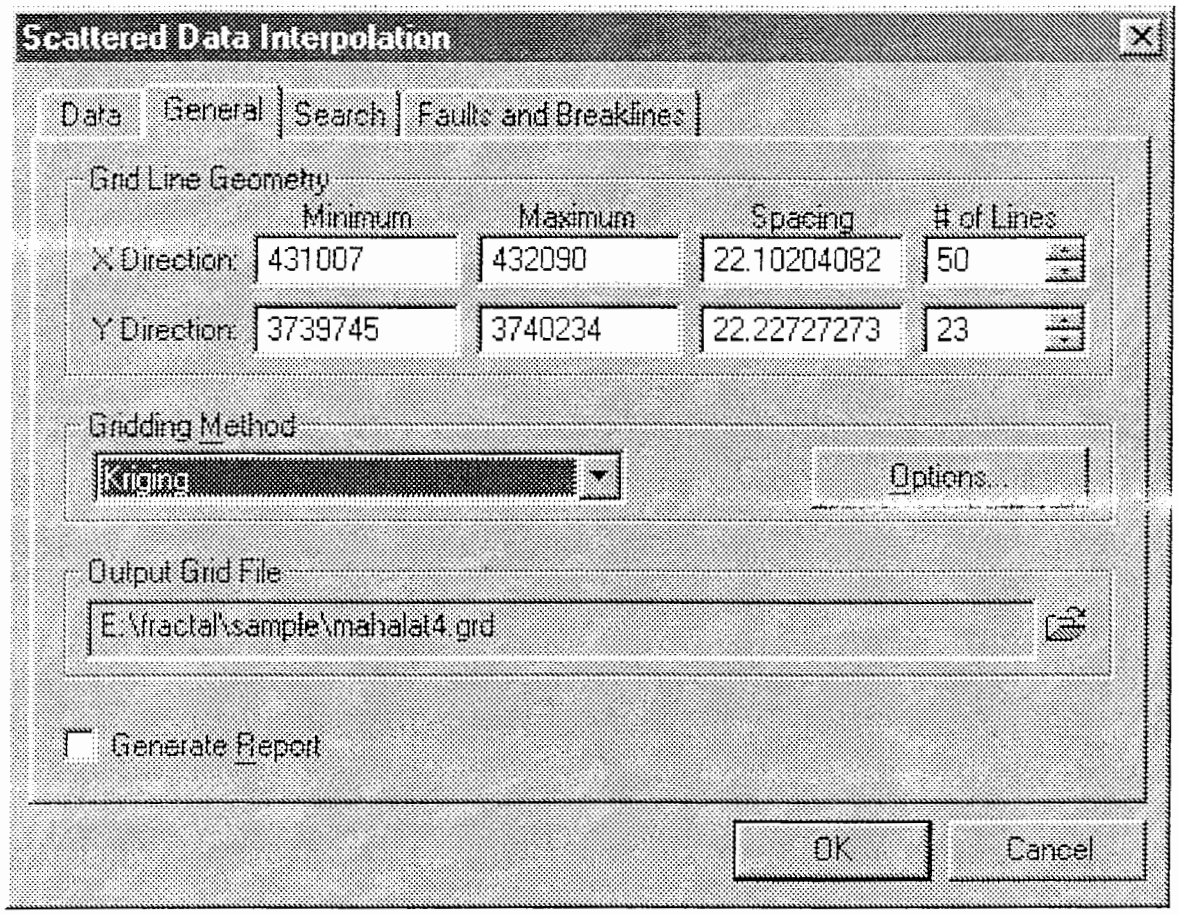
۶-۲ نحوه ارسال داده‌ها به نرم‌افزارهای FM و FS

ابتدا نرم‌افزار سورفر فایل ورودی داده‌ها را تبدیل به فایل‌های با پسوند grd. و به صورت متنی می‌نماید. نحوه کار بدین صورت است که ابتدا از منوی grid گزینه data را انتخاب نموده و فایل داده‌های اصلی بعنوان فایل ورودی معرفی می‌شوند. در پنجره Scattered Data Interpolation تعداد خطوط شبکه‌بندی و فاصله بین خطوط شبکه‌بندی در محدوده x وy و همچنین پارامترهای دیگری نظیر روش شبکه بندی را می‌شود تتظیم کرد (شکل۶-۱).

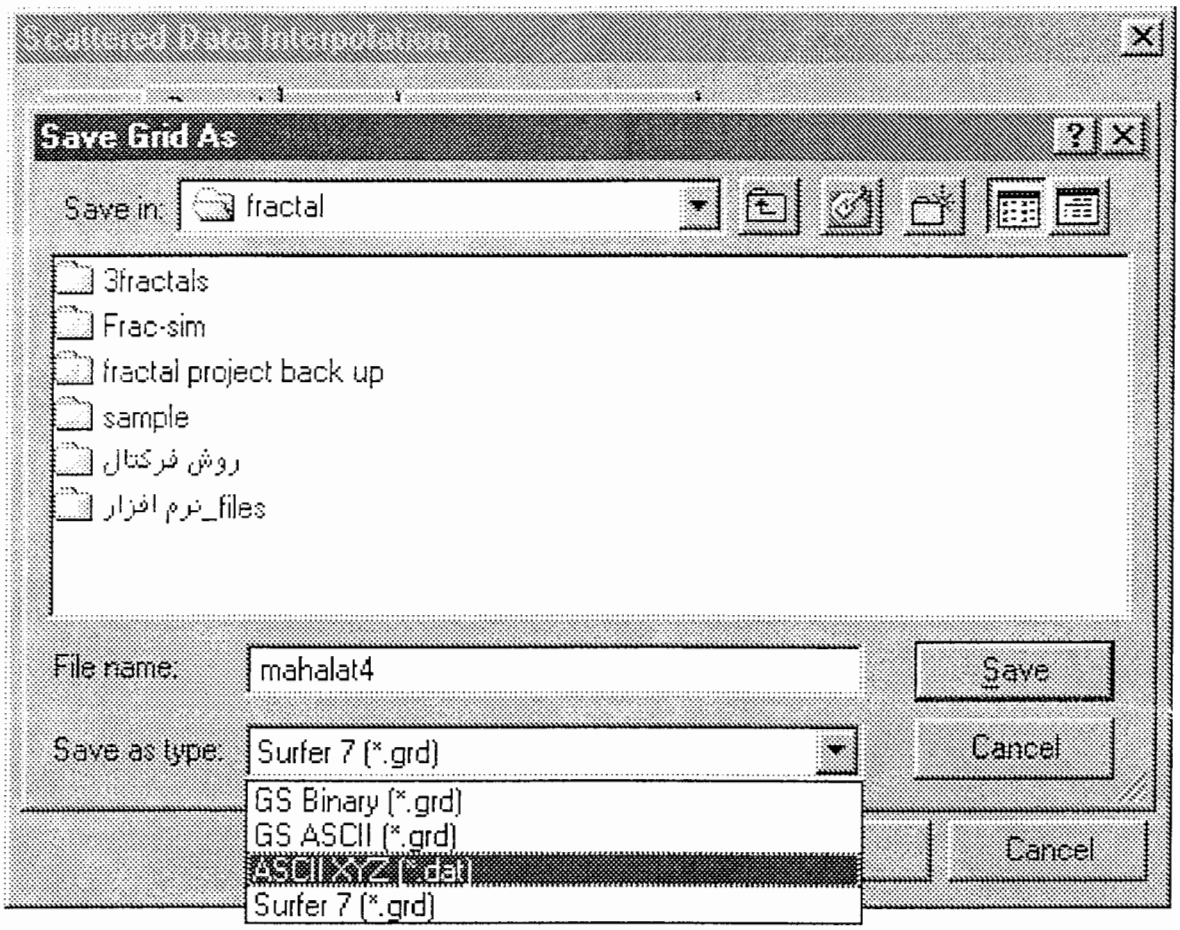
سپس در پانل Out Put Grid File نام فایل خروجی را تعیین نموده و در کادرSave Grid as در قسمت Save as type گزینه ASCII XYZ انتخاب می‌شود (شکل ۶-۲). در نتیجه خروجی‌ها به صورت متنی ذخیره می‌شوند.

از منوی File و توسط گزینه Open فایل متنی شبکه بندی شده بازخوانی می‌شود. در اینجا می‌بایست داده‌ها به طریق خاصی مرتب‌سازی گردند. به این نحو که ابتدا بر حسب ستون y ها و سپس بر حسب ستون xها عمل مرتب‌سازی انجام می‌گیرد. به این منظور از منوی Data گزینه

Sort را انتخاب کرده (شکل ۶-۳) و بدینترتیب پنجره Sort باز می‌شود. روش کار به این ترتیب است که در پانل Sort Frist By به ترتیب گزینه‌های مربوط به محور yها و سپس محور xها انتخاب می‌شوند (شکل ۶-۴).

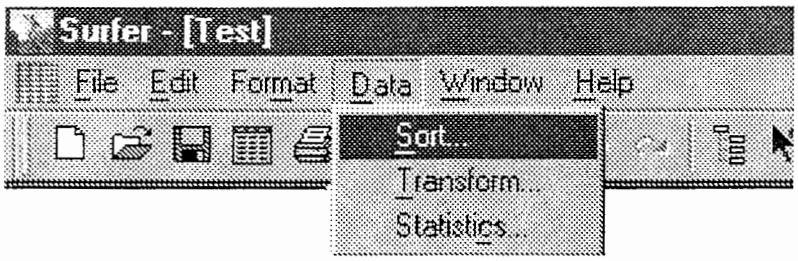


شکل ۶-۱ شمای گرافیکی پنجره Scattered data interpolation

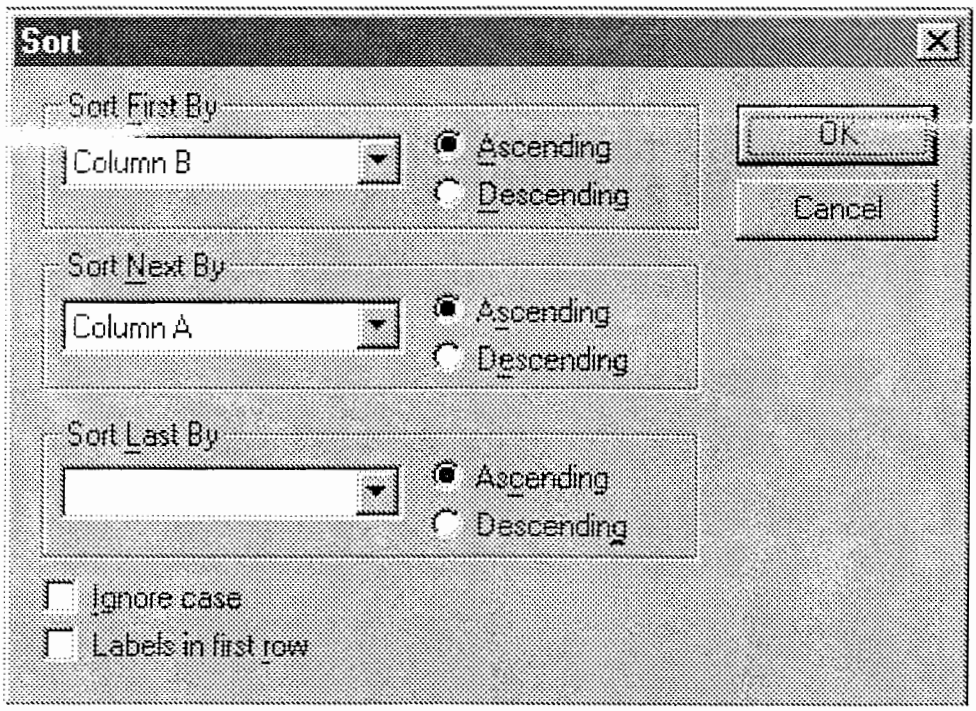


شکل ۶-۲ شمای گرافیکی پنجره Save Grid As

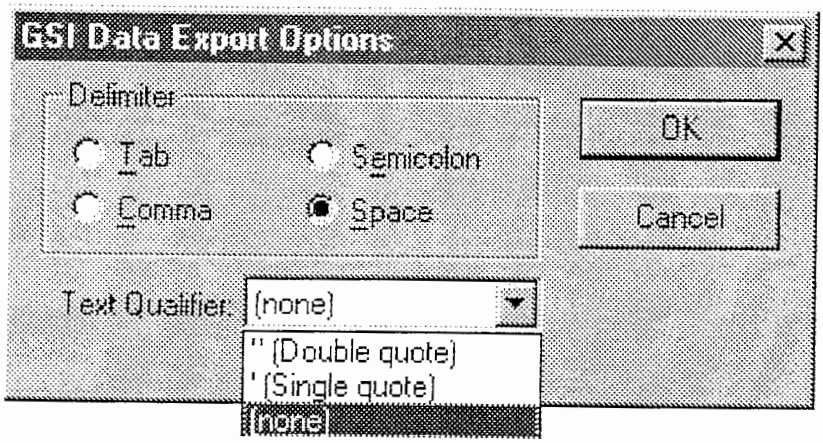
هر دو پارامتر x و y را بصورت افزیشی مرتب نموده و فایل را ذخیره می‌کنیم. برای این کار ابتدا از منوی فایل گزینه Save as انتخاب می‌شود و پنجره GSI باز شده و پارامترهای Delemiter و Text Qualifier به ترتیب بر روی Space و None تنظیم می شوند (شکل ۶-۵).



شکل ۶-۳ نمایش نحوه انتخاب دستور Sort از منوی Data



شکل ۶-۴ نحوه تنظیمات پنجره Sort



شکل ۶-۵ تنظیمات پنجره Gsi

۶-۳ طریقه کلی استفاده از نرم‌افزار FM

پس از اجرای برنامه از منوی فایل گزینه Open Data انتخاب می‌شود (شکل ۶-۶). فایل ورودی تهیه شده، به عنوان ورودی نرم‌افزار انتخاب می‌شود. پس از مدت کوتاهی، کلیه داده‌ها در یک جدول مربعی نمایش داده می‌شوند. سطر اول نشان دهنده مقادیر محور xها و ستون اول نشان دهنده مقادیر محور yها می‌باشند. در مابقی خانه‌های جدول مقادیری از ارتفاع نقاط شبکه‌بندی به ترتیب مختصات x و y آنها نمایش داده می‌شود (شکل ۶-۷).

در این مرحله امکان محاسبه بعد فراکتال به سه روش پرگار تقسیم سیار، تغییرنما و پرگار تقسیم سیار دوبعدی وجود خواهد داشت (شکل ۶-۸).

۶-۳-۱ نحوه محاسبه بعد فراکتال به روش پرگار تقسیم سیار

از منوی Run در شکل ۶-۸ گزینه Floating Dividing Scale انتخاب می‌شود. کادر Loop Number باز شده، مقدار Loop را که در واقع همان تعداد مراحل نصف شدن گام پرگار می‌باشد وارد می‌شود. این مقادیر با توجه به تعداد داده‌ها و همچنین مقدار پارامتر x Spacing بستگی مستقیم دارد. عموماً برای این مقدار مقادیر بین ۱۵ تا ۲۵ انتخاب می‌گردند (شکل ۶-۹).

سپس از منوی Run (شکل ۶-۸) گزینه Graph of Floating Dividing Scale را انتخاب کرده بدینترتیب پنجره Graph باز می‌گردد.

شکل ۶-۶ شمای ظاهری برنامه قبل از ورود اطلاعات

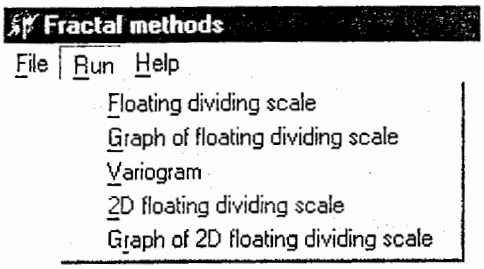
Fractal method E:\test1.dat

File Run Help

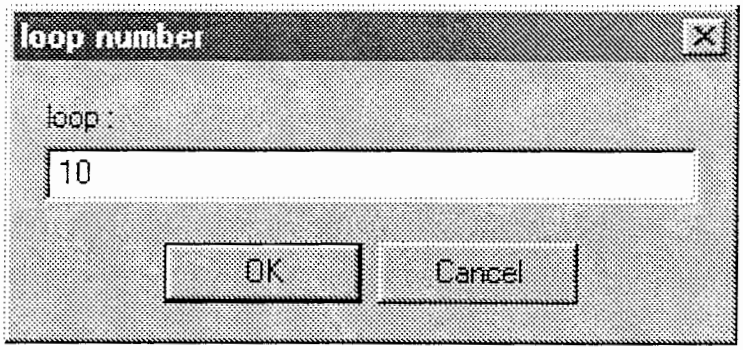
Y \ X	431007	431029.10204082	431051.20408163	431073.30612245	431095.40816327	431117.510
3739745	1975.5465922848	1976.2998045408	1977.3551188123	1978.5950611279	1979.9095874756	1981.21898
3739767.2272727	1975.1468508672	1976.0367773571	1977.335201861	1978.8213931729	1980.3452056471	1981.82339
3739789.4545455	1974.6953970128	1975.8333699796	1977.4959895647	1979.2670359529	1980.9980786167	1982.63101
3739811.6818182	1974.1833956235	1975.9784100469	1978.0618093677	1980.0584936465	1981.9369792024	1983.67966
3739833.9090909	1975.6995886106	1977.2502743584	1979.278516286	1981.2870605908	1983.198305447	1984.98266
3739856.1363636	1978.1646915426	1979.368197305	1981.0639547074	1982.912104367	1984.752913823	1986.51485
3739878.3636364	1980.5372436519	1981.6432599343	1983.1184175143	1984.7870403731	1986.5135191586	1988.21406
3739900.5909091	1982.8058370845	1983.8852578446	1985.2359511248	1986.7627070095	1988.3726518841	1989.99556
3739922.8181818	1984.9596135816	1986.0374909436	1987.3138385015	1988.7317283526	1990.231787572	1991.76692
3739945.0454545	1986.9887821534	1988.0748142784	1989.3003398895	1990.6253710259	1992.0122435434	1993.43823
3739967.2727273	1988.8845704749	1989.9841458122	1991.1703331796	1992.4053089932	1993.6601408356	1994.93198
3739989.5	1990.6378420091	1991.7585480674	1992.9164807831	1994.0614539288	1995.154864851	1996.20023
3740011.7272727	1992.2354902175	1993.3913132279	1994.5427067739	1995.6120277215	1996.5226153486	1997.25106
3740033.9545455	1993.6543522805	1994.8651149793	1996.0498565773	1997.0942229279	1997.8489536978	1998.18839
3740056.1818182	1994.8547143991	1996.1354221917	1997.4049251299	1998.5223414351	1999.2491038931	1999.30486

progress /

شکل ۶-۷ شمای ظاهری برنامه پس از ورود اطلاعات



شکل ۶-۸ نحوه انتخاب دستورات مختلف محاسبه و نمایش مقادیر بعد فراکتال



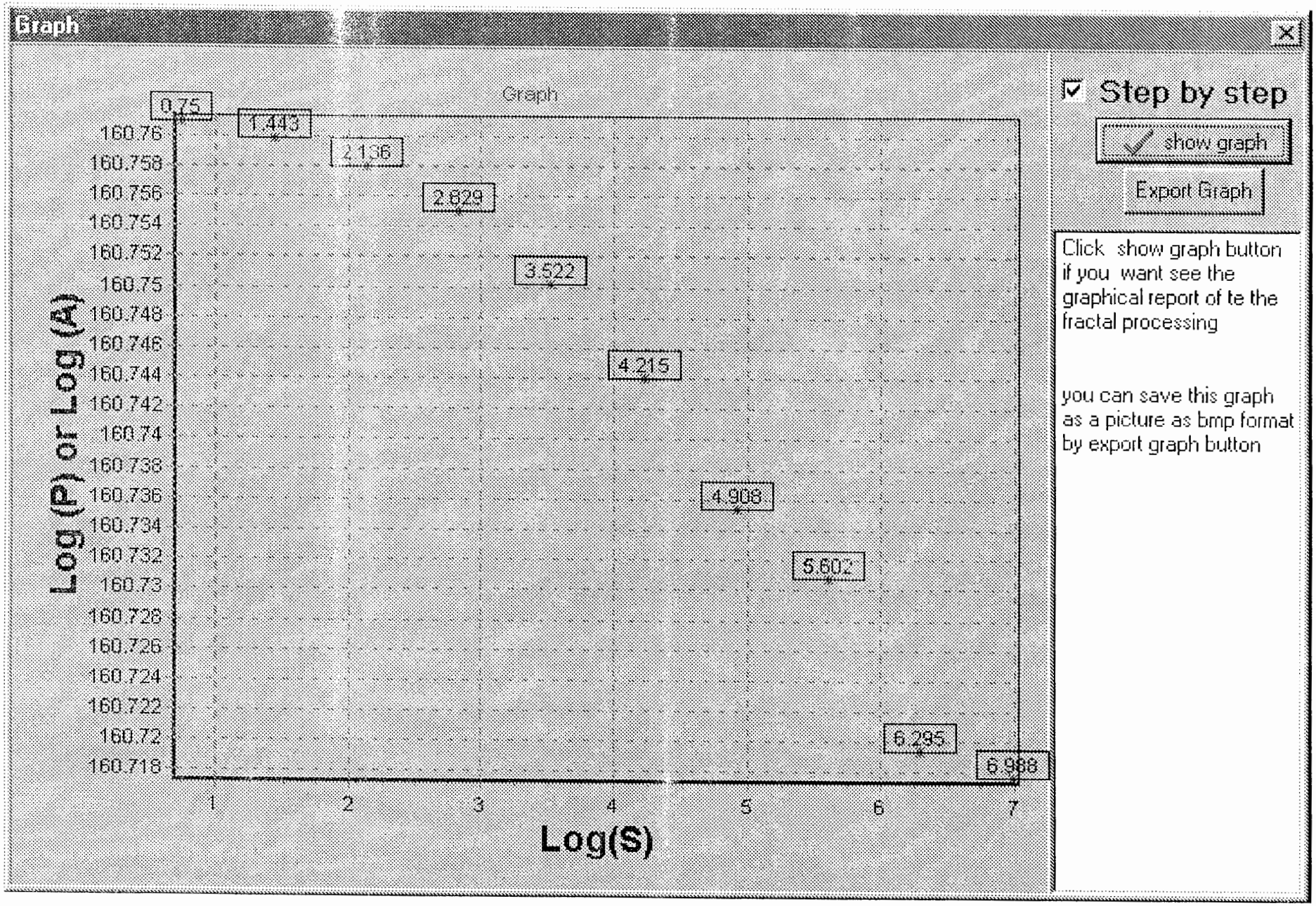
شکل ۶-۹ شمای کلی پنجره Loop Number

جهت مشاهده نمودار لگاریتم محیط در برابر لگاریتم طول پرگار، می‌بایست بر روی دکمه Show graph کلیک نمود (شکل ۶-۱۰).

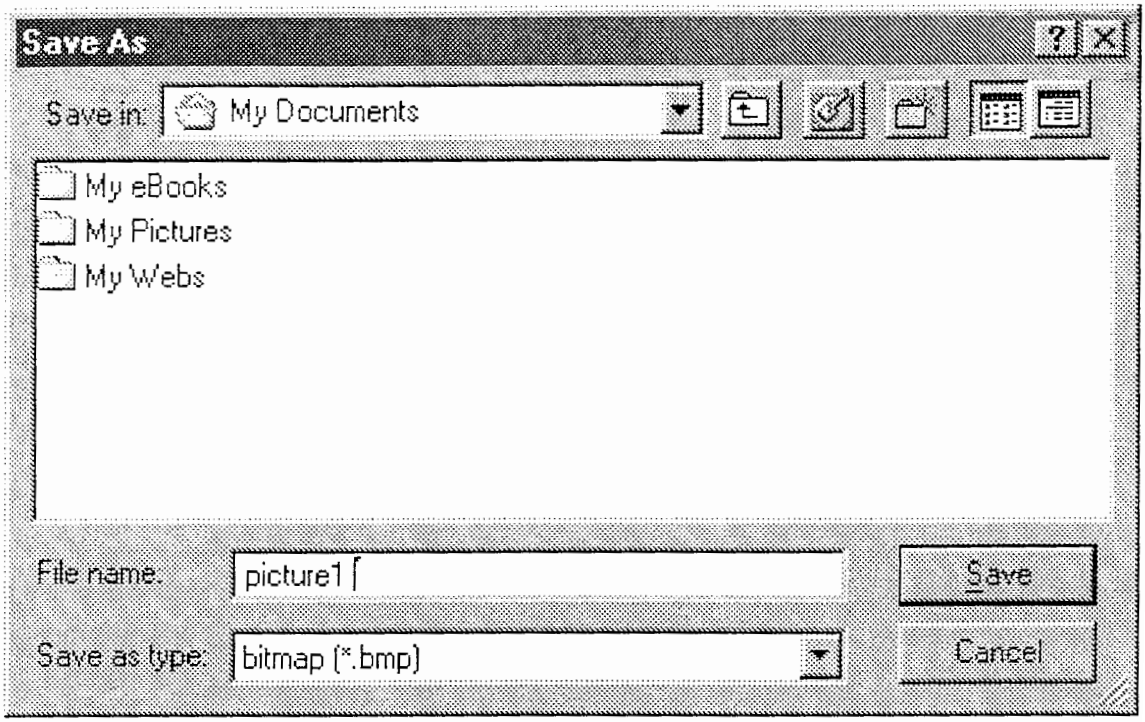
۶-۳-۲ محاسبه بعد فراکتال به روش تغییرنما

برای محاسبه بعد فراکتال به روش تغییرنما، پس از ورود داده‌ها به نرم‌افزار، از منوی Run گزینه Variogram انتخاب می‌شود. بلافاصله فرآیند محاسبه بعد فراکتال آغاز می‌گردد. در پایان مقدار تغییرنما (b) در یک کادر نمایش پیغام مشاهده می‌گردد. در شکل ۶-۱۱ نحوه ذخیره کردن نتایج و در شکل ۶-۱۲ شمایی از کادر نمایش پیغام آورده شده است. در نهایت بعد فراکتال سطح با استفاده از رابطه ۶-۱ محاسبه می‌شود.

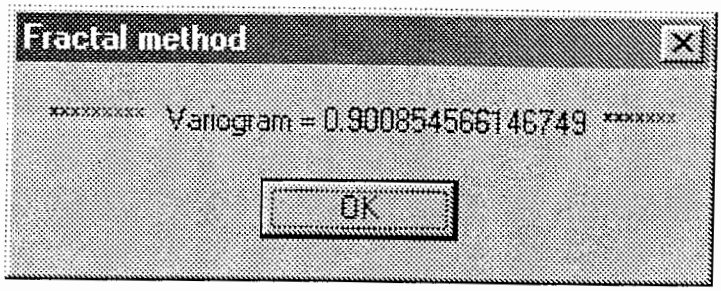
$$D = 2 + 1 - b \qquad (۱-۶)$$



شکل ۶-۱۰ نمایی از پنجره Graph



شکل ۶-۱۱ نحوه ذخیره کردن نتایج



شکل ۶-۱۲ نمایش مقدار تغییرنمای سطح که به توسط نرم‌افزار محاسبه شده است

۶-۳-۳ محاسبه بعد فراکتال به روش پرگار تقسیم سیار دو بعدی

برای محاسبه بعد فراکتال داده‌ها به روش پرگار تقسیم سیار دو بعدی، پس از ورود داده‌ها به نرم‌افزار از منوی Run گزینه 2Dfloyting Diviping Scale انتخاب می‌شود. در ابتدا تعداد مراحل دو برابر شدن گام از کاربر پرسیده می‌شود. پس از پایان فرآیند محاسبه، بعد از منوی Run گزینه Graphe 2Dfloyting Diviping Scale را از شکل ۶-۸ انتخاب نموده و بدینترتیب پنجره نمایش گراف ظاهر می‌گردد.

در پنجره نمایش گراف برای مشاهده نتایج حاصله بر روی دستور Show Graph کلیک کرده و لگاریتم مساحت کلی سطح به ازای لگاریتم مساحت گام به عنوان مقدار متغیر (b) در نظر گرفته می‌شود. سپس بعد فراکتال توسط رابطه ۶-۲ بدست می‌آید.

$$D = 2 + b \qquad (۶-۲)$$

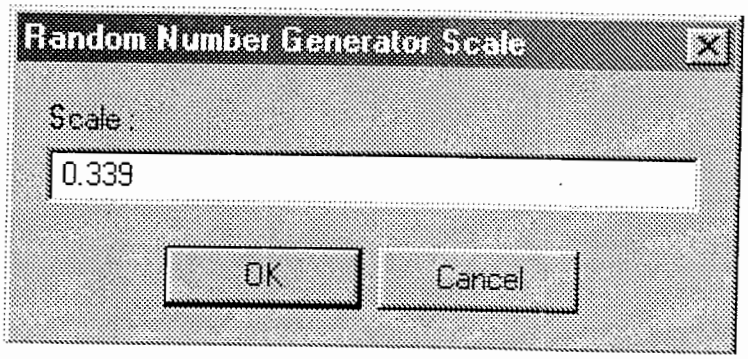
چنانچه نیاز به تهیه خروجی گرافیکی از نمودار حاصله باشد با کلیک بر روی دستور Export Graph میتوان فایل‌هایی با فرمت bmb. تهیه نمود.

۶-۴ نحوه استفاده ازنرم‌افزار Fractal Simulation

پس از اجرای برنامه از منوی File، گزینه Open را انتخاب نموده و فایل ورودی را معرفی می‌کنیم. فرمت نمایش اطلاعات در اینجا نیز شبیه نرم‌افزار FM می‌باشد.

پس از نمایش داده‌ها در جدول، مقدار تغییرنمای داده‌های ورودی مانند شکل ۶-۱۲ در یک کادر نمایش داده می‌شود.

حال می‌بایست عمل تخمین داده‌های میانی به روش جابجایی نقطه میانی انجام گیرد. برای این کار از منوی Run گزینه Process انتخاب می‌گردد. کادر Random number generator scale باز می‌شود (شکل ۶-۱۳).



شکل ۶-۱۳ شمایی از کادر Random number generator scale

در اینجا باید عددی را بعنوان ضریب اختصاص داده شده به اعداد تصادفی تولید شده برای اختصاص به داده‌های میانی وارد کرد. این عدد باید به گونه‌ای انتخاب شود که پس از انجام عمل درون‌یابی بعد فراکتال داده‌های حاصل از یک بار جابجایی نقطه میانی با بعد فراکتال داده‌های ورودی یکسان باشد.

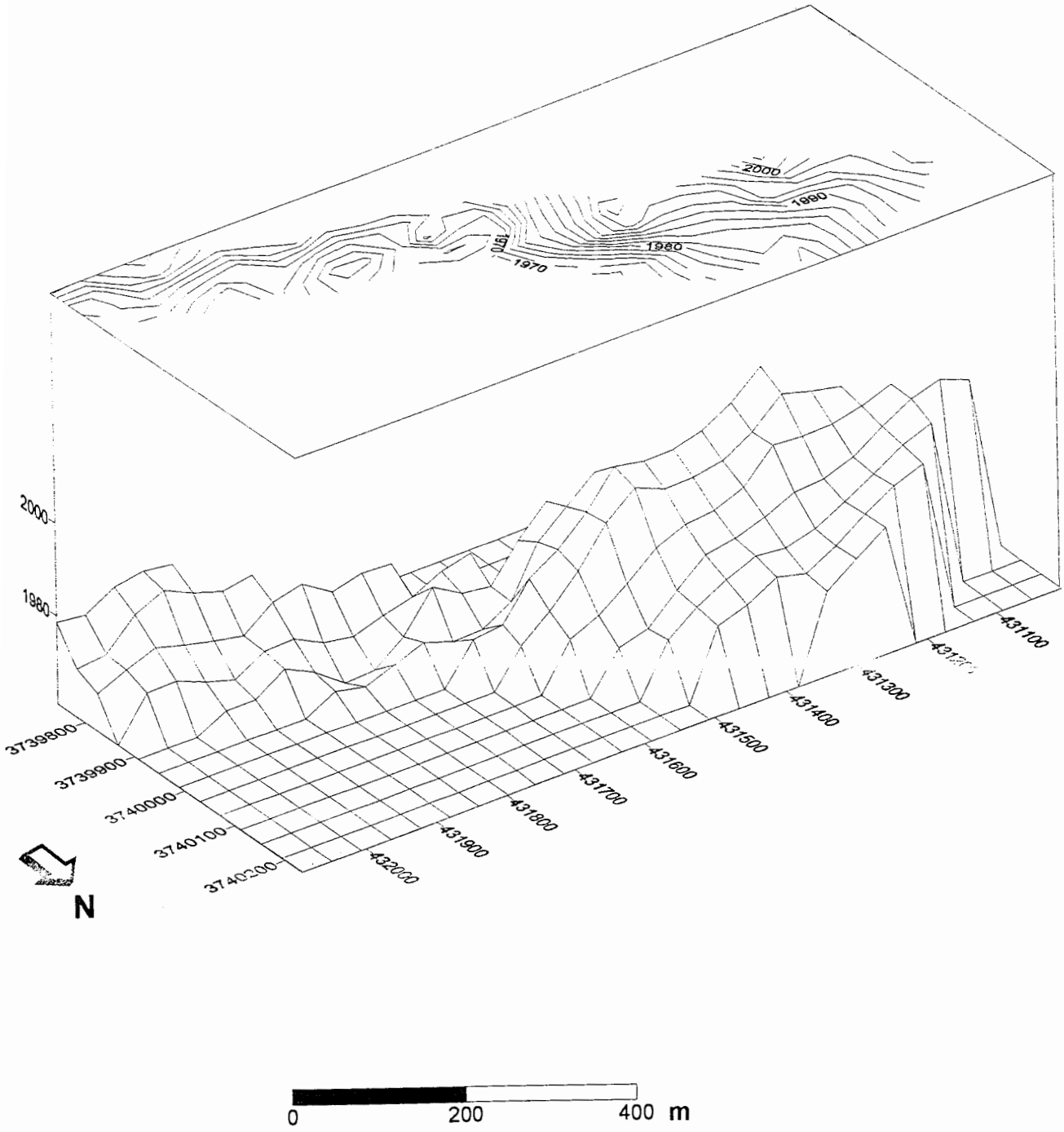
بدین منظور می‌بایست عموماً عددی را بین صفر تا یک به متغیر مربوطه اختصاص داد. هر چه این مقدار به صفر نزدیک‌تر شود مقدار تغییرنما بزرگتر خواهد بود و هر چه این مقدار به یک نزدیکتر شود، مقدار تغیرنما کوچکتر خواهد بود. مقدار مناسب برای این ضریب را می‌توان به روش جستجوی دودویی با دقت سه رقم اعشار با انجام حدوداً ۵ تا ۱۰ بار تکرار بدست آورد. در صورت نیاز نتایج محاسبات توسط دستور Save as از منوی فایل قابل ذخیره‌سازی می‌باشد. همچنین می‌توان توسط دستور Save white noise از منوی فایل، ماتریس اعداد تصادفی تولید شده را ذخیره نمود. در جدول ۶-۱ مقادیر تغییرنما و مقادیر بعد فراکتال نقاط اطلاعات توپوگرافی منطقه محلات ۴ بعنوان نمونه آورده شده است. بدیهی برای کلیه مناطقی که در آنها نقشه‌برداری صورت می‌گیرد، همچون فایل مذکور می‌توان شبکه اطلاعاتی را چگال نمود بگونه‌ایکه بعد فراکتال شبکه چگال شده و شبکه اولیه یکسان باقی بماند. همچنین جهت مقایسه در اشکال ۶-۱۴ الی ۶-۱۷ نقشه‌های توپوگرافی و سه‌بعدی منطقه محلات ۴ آورده شده است. در شکل ۶-۱۴ نقشه کریجینگ شده با طول شبکه ۴۱ متر رسم شده است. لازم به ذکر است که مساحت منطقه مطالعاتی حدود ۴,۵ هکتار و تعداد نقاط برداشت شده ۱۱۲ نقطه بوده است. لذا طول شبکه اولیه از حاصل تقسیم مساحت منطقه بر تعداد داده‌ها بدست آمده است. همانگونه که در جدول ۶-۱ ملاحظه می‌شود بعد فراکتال سطح اولیه که به روش تغییرنما محاسبه شده برابر ۲,۱۷۲ است. در شکل ۶-۱۵ نقشه توپوگرافی و سه‌بعدی داده‌هایی رسم شده که حاصل یک بار جابجایی نقطه میانی شدن سطح اولیه هستند، بگونه‌ایکه طول شبکه به حدود ۲۰ متر کاهش یافته اما بعد فراکتال سطح تغییر نکرده است. البته این موضوع در جدول ۶-۱ قابل مشاهده است. در شکل ۶-۱۶ نقشه توپوگرافی و سه‌بعدی سطح حاصل از جابجایی نقطه میانی شدن سطح قبلی ترسیم شده است. در این فایل طول شبکه به حدود ۱۰ متر کاهش یافته است. همانگونه که در جدول ۶-۱ ملاحظه می‌شود بعد فراکتال این سطح نیز با بعد فراکتال سطوح قبل برابر و مساوی ۲,۱۷۲ است. بنابراین در واقع هنر تکنیک بکار برده شده، در شبیه‌سازی سطوحی است که بعد فراکتال آنها تغییر نمی‌کند. به عبارت دقیقتر تخمینگر فراکتال یا جابجایی نقطه میانی سطوحی را در نقشه‌برداری گسترش می‌دهد که به لحاظ آماری معتبر هستند. جهت مقایسه در شکل ۶-۱۷ نقشه توپوگرافی و سه بعدی اطلاعات محلات ۴ با تخمینگر کریجینگ و طول شبکه ۱۰ متر ترسیم شده است. همانگونه که در جدول ۶-۱ ملاحظه می‌شود بعد فراکتال این سطح به ۲,۰۵۶ کاهش یافته که در واقع این موضوع با خصلت تخمینگر کریجینگ که نرم‌سازی است مطابقت دارد.

جدول ۶-۱ نتایج چگال کردن سطوح و بعد فراکتال حاصل از آنها

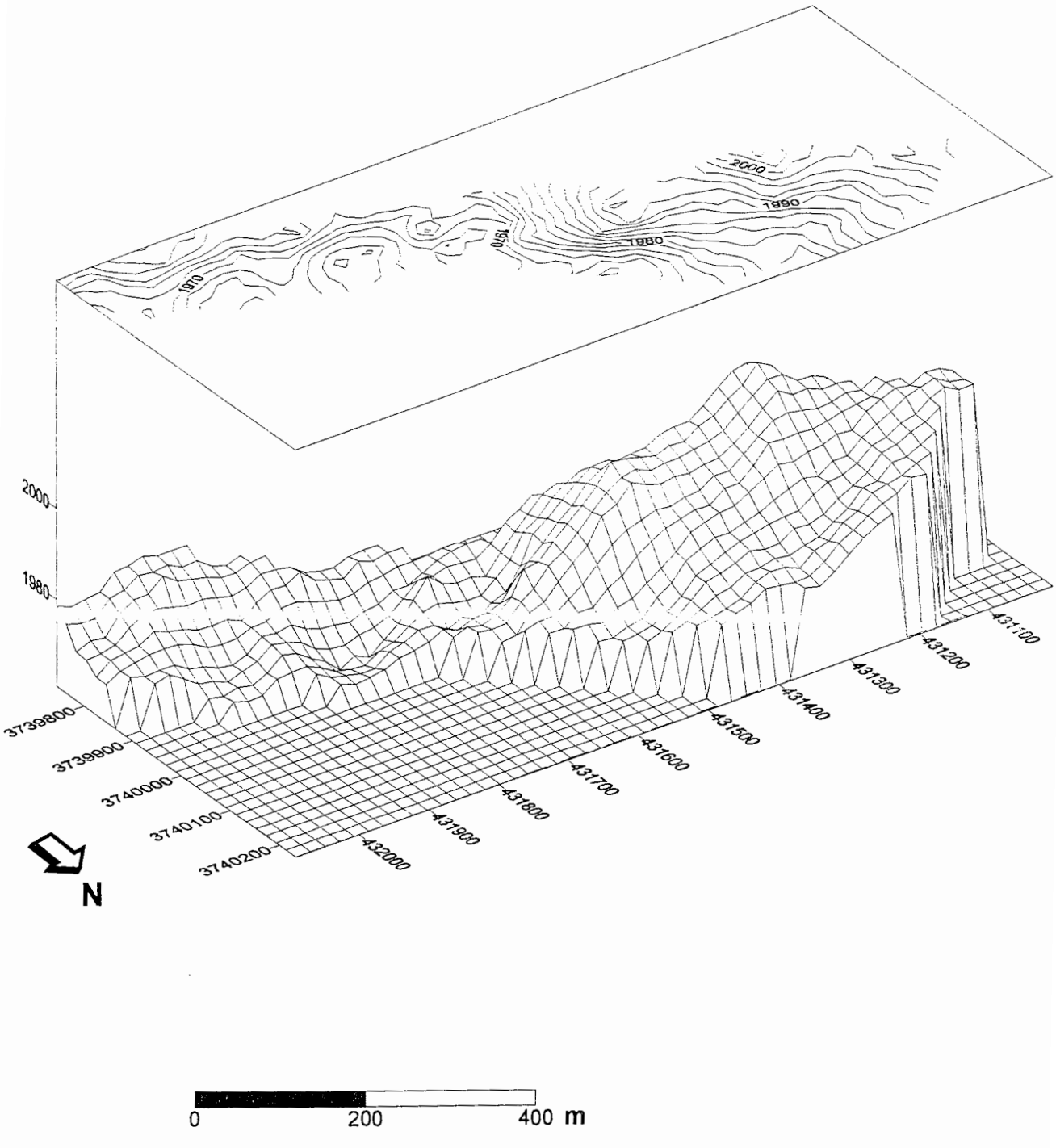
شماره شکل	فواصل خطوط شبکه	پردازشگر	مقدار تغییرنما	بعد فراکتال
۶-۱۴	۴۱	کریجینگ	۰,۸۲۲	۲,۱۷۲
۶-۱۵	۲۰	جابجایی نقطه میانی	۰,۸۲۸	۲,۱۷۲
۶-۱۶	۱۰	جابجایی نقطه میانی	۰,۸۲۸	۲,۱۷۲
۶-۱۷	۱۰	کریجینگ	۰,۹۴۴	۲,۰۵۶

۶-۵ توضیحات تکمیلی درباره نرم‌افزار Fractal Simulation (FS)

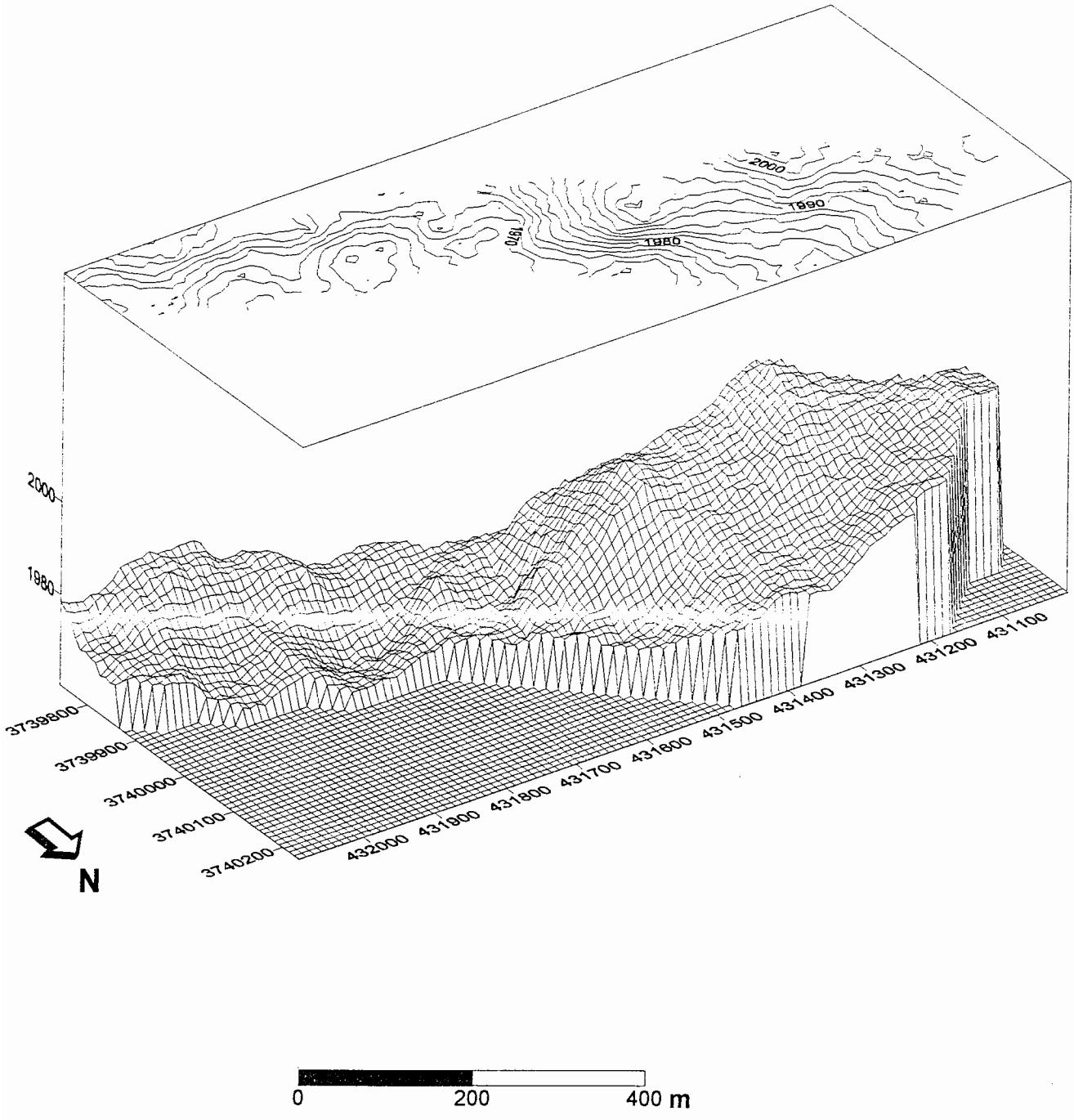
در این نرم‌افزار فرآیند جابجایی نقطه میانی به شرح زیر انجام می‌گیرد. در مرحله اول داده‌هایی که شماره ستون و سطر مربوط به آنها هر دو زوج است مورد بررسی قرار می‌گیرد، بطوریکه مقدار هر داده جمع میانگین چهار داده مجاور به علاوه جبرانی تصادفی متناسب با طول شبکه بدست می‌آید. این جبرانی تصادفی حاصل ضرب عدد تصادفی گوسی در مقیاس (Scall) مربوط به طول شبکه بدست می‌آید. در مرحله بعد، نوبت به داده‌هایی می‌رسد که فقط یکی از شماره‌های مربوط به سطر و ستون آنها زوج است و دیگری فرد می‌باشد. شبکه مربوط به این داده‌ها نسبت به شبکه اولیه دارای چرخش ۴۵° می‌باشد. لذا مقدار آن از حاصل جمع میانگین چهار همسایه مجاور و $\frac{\sqrt{2}}{2}$ برابر جبرانی تصادفی متناسب با طول شبکه بدست می‌آید. در مرحله آخر نقاطی از سطرها و ستون‌های لبه شبکه که می‌بایست درون‌یابی شوند از حاصل جمع میانگین سه همسایه مجاور آنها و جبرانی تصادفی متناسب با طول شبکه محاسبه می‌شوند.



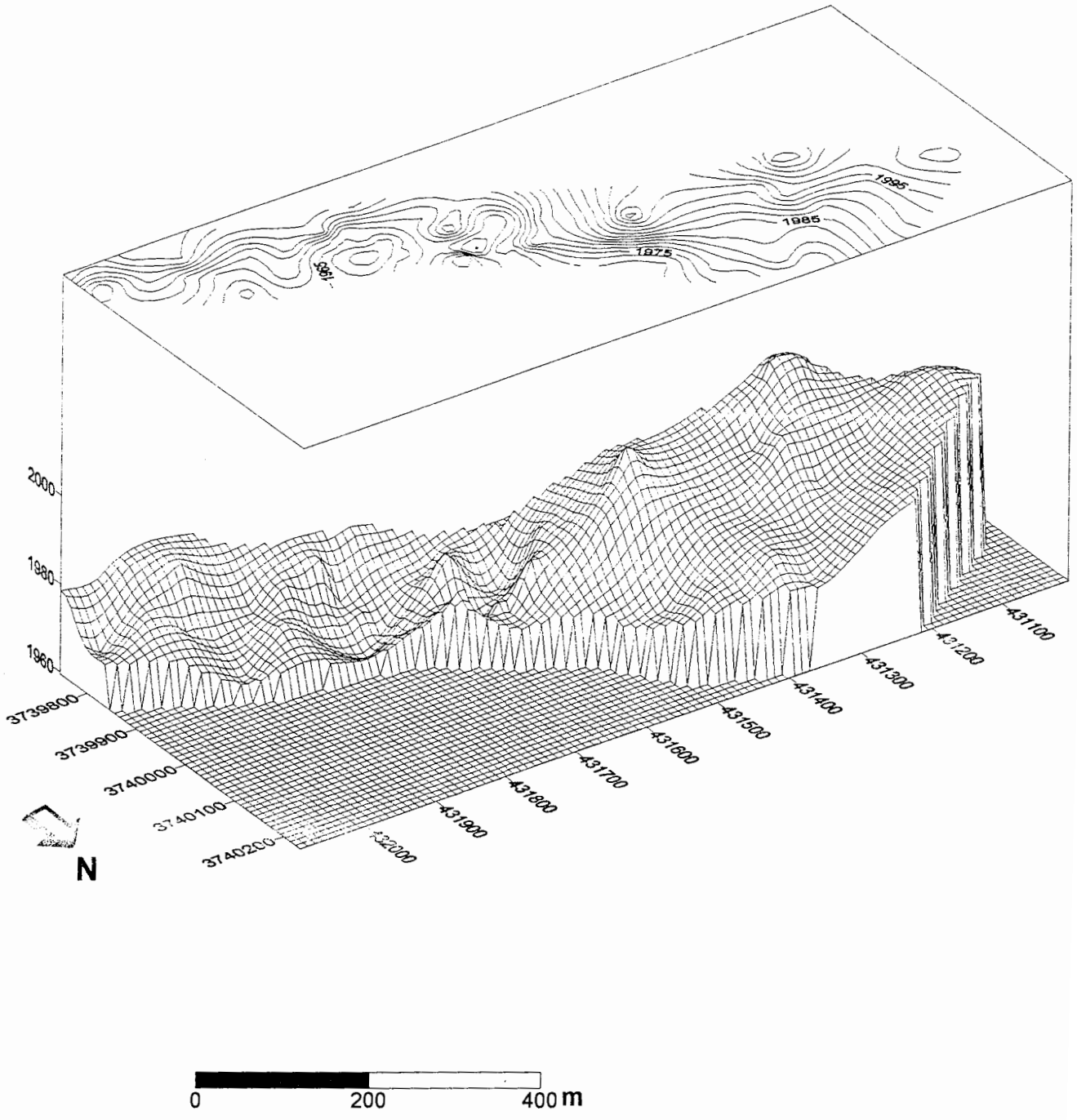
شکل ۶-۱۴ نقشه توپوگرافی و سه بعدی حاصل از کریجینگ داده‌های منطقه محلات ۴ با فواصل خطوط شبکه ۴۰ متر



شکل ۶-۱۵ نقشه توپوگرافی و سه بعدی حاصل از جابجایی نقطه میانی داده‌های منطقه محلات ۴ با فواصل خطوط شبکه ۲۰ متر



شکل ۶-۱۶ نقشه توپوگرافی و سه بعدی حاصل از جابجایی نقطه میانی داده‌های منطقه محلات ۴ با فواصل خطوط شبکه ۱۰ متر



شکل ۶-۱۷ نقشه توپوگرافی و سه بعدی حاصل از کریجینگ داده‌های منطقه محلات ۴ با فواصل خطوط شبکه ۱۰ متر

۷

نتیجه‌گیری و پیشنهادات

۷-۱ نتیجه‌گیری

در هندسه فراکتال بحث از خطوط و سطوحی است که بعد عدد صحیح یک و دو ندارند. به عبارت دیگر یک خط شکسته، بعدی بین یک و دو و یک سطح منحنی، بعدی بین دو و سه دارد. بنابراین در این تحقیق سعی بر آن بوده که بعد فراکتال سطوح طبیعی را هنگام نقشه‌برداری محاسبه کرده و سپس از تخمینگری با منطق فراکتال به نام جابجایی نقطه میانی جهت چگال کردن اطلاعات استفاده شود، بگونه‌ایکه در نهایت بعد فراکتال نقشه‌ای که بعنوان نقشه توپوگرافی سطح ترسیم می‌شود برابر با بعد فراکتال سطح اولیه باشد. در این راستا نرم‌افزاری تحت عنوان روشهای فراکتالی[1] طراحی و با زبان برنامه‌نویسی دلفی تحت سیستم عامل ویندوز نوشته شد که به سه روش پرگار تقسیم سیار، پرگار تقسیم سیار دوبعدی و تغییرنما، مقدار بعد فراکتال خطوط و سطوح را محاسبه می‌نماید. بدینترتیب بعد فراکتال کلیه اشکال قابل محاسبه هستند.

سپس نرم‌افزاری تحت عنوان شبیه‌ساز فراکتال[2] طراحی و نوشته شد که به توسط الگوریتم جابجایی نقطه میانی از روی اطلاعات اولیه، سطوحی را شبیه‌سازی می‌کند که بعد فراکتال آنها با بعد فراکتال طبیعی سطوح برابر است. در فصول گذشته نتیجه عملکرد الگوریتمهای مذکور و توانمندی روش فراکتال در شبیه‌سازی ارائه گردید.

در نهایت ملاحظه می‌شود که به توسط روش فراکتال، سطوحی را می‌توان شبیه‌سازی کرد که به لحاظ آماری کاملاً شبیه به سطوح اولیه هستند. البته همچنین در فصول گذشته ثابت گردید که تخمینگرهای دیگر و در راس آنها کریجینگ، قابلیت شبیه‌سازی سطوح معتبر آماری را ندارند، هر چند که ممکن است اینگونه تخمینگرها کمترین واریانس تخمین را نیز ارائه دهند.

---

[1] -Fractal Methods (FM)

[2] - Fractal Simulation (FS)

۷-۲ پیشنهادات

با توجه به نتیجه مطالعات صورت گرفته فضای بسیار وسیعی در مقابل دید برای مطالعات آتی گشوده گردید که تعدادی به عنوان پیشنهاد معرفی می‌گردند.

الف – بدون شک از منطق فراکتال جهت ترسیم منحنی‌های هم‌عیار، هم‌ضخامت و ... در فعالیتهای معدنکاری و همچنین منحنی‌های هم‌گرانی، هم‌مغناطیس و ... در برداشتهای ژئوفیزیکی می‌توان بهره جست. لذا توصیه می‌شود کاربردهای منطق فراکتال در وادی‌های مذکور مطالعه گردد.

ب – یقیناً یکی از نقاط ضعف مطالعات فعلی مشخص نکردن خطای تخمین است. البته در این طرح خطای منطق فراکتال نسبت به روش کریجینگ مورد مطالعه قرار گرفته است و میانگین و واریانس اختلاف بین مقادیر تخمین زده شده به روشهای فراکتال و کریجینگ به ترتیب برابر صفر و کوچکتر از یک بوده‌اند. ولی بدون شک مطالعه دقیقتر و کمی نمودن خطای تخمینگر فراکتال ضروری است.

ج – البته الگوریتم جابجایی نقطه میانی، الگوریتم مناسبی و قوی جهت شبیه‌سازی سطوح است. اما مطالعه و ارائه نرم‌افزار برای الگوریتمهای احتمالی دیگر و یا ارائه الگوریتمهای جدید نیز مفید خواهد بود.

پیوست اول

کدهای اصلی فرم اصلی نرم‌افزار

Fractal Methods
(FM)

```
    n1: TMenuItem;
    Open1: TMenuItem;
    New1: TMenuItem;
    Help1: TMenuItem;
    About1: TMenuItem;
    openxyzfile: TOpenDialog;
    stgrid: TStringGrid;
    Panel1: TPanel;
    Run1: TMenuItem;
    floatingdividingscale: TMenuItem;
    graphfds: TMenuItem;
    variogram: TMenuItem;
    status: TStatusBar;
    N2Dfloatingdividingscale: TMenuItem;
    graph2dfds: TMenuItem;
    Opennoise1: TMenuItem;
    procedure Open1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure variogramClick(Sender: TObject);
    procedure floatingdividingscaleClick(Sender: TObject);
    procedure graphfdsClick(Sender: TObject);
    procedure N2DfloatingdividingscaleClick(Sender: TObject);
    procedure graph2dfdsClick(Sender: TObject);
    procedure Opennoise1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

uses Unit2;

{$R *.DFM}

constructor tdatagrid.create;
      begin
      loop:=0;
      x_line_no:=0;
      y_line_no:=0;
      z_line_no:=0;
      x_spacing:=0;
      y_spacing:=0;
      lag:=10;
      lag_zero:=10;
      x_start:=0;
```

```
    z_start:=0;
    x1:=0;
    z1:=0;
    x2:=0;
    z2:=0;
    setlength(data,0,0);
    setlength(perimeter_matrix,0,0,0);
    end;


procedure tdatagrid.read_data;
    var
    f1:textfile;
    ch:char;
    ch1:shortstring;
    i:integer;
    j:integer;
    begin
    if form1.openxyzfile.Execute then
    begin
    setlength(data,0,0);
        for i:=0 to form1.stgrid.ColCount do
            for j:=0 to form1.stgrid.RowCount do
            begin
            form1.stgrid.Cells[i,j]:='';
            end;
        form1.stgrid.ColCount:=1;
        form1.stgrid.RowCount:=1;
        form1.stgrid.FixedCols:=0;
        form1.stgrid.FixedRows:=0;
        i:=0;
        assignfile(f1,form1.openxyzfile.FileName);
        reset(f1);
        while not eof(f1) do
            begin
            i:=i+1;
            readln(f1);
            end;
        setlength(data,3,i);
        z_line_no:=i;
        i:=0;
        j:=0;
        ch1:='';
        reset(f1);
        while not eof(f1) do
            begin
            read(f1,ch);
            if ((ch<>' ')and(ch<>#13))then
                begin
                if (ch<>#10)then
```

```
                    begin
                    ch1:=ch1+ch;
                    end;
                if (ch=#10)then
                    begin
                    i:=i+1;
                    j:=0;
                    end;
                end;
            if ((ch=' ')or(ch=#13))then
                begin
                data[j][i]:=strtofloat(ch1);
                j:=j+1;
                ch1:='';
                end;
            end;
        closefile(f1);
        i:=0;
    while data[1,i]=data[1,0] do // # of line X lineno//
        begin
        i:=i+1;
        end;
    x_line_no:=i;
    y_line_no:=round(z_line_no/x_line_no);
    for i:=0 to z_line_no-1 do
        begin
        form1.stgrid.Cells[((i) mod x_line_no)+1,0]:=floattostr(data[0,i]);
        end;
    for i:=0 to z_line_no-1 do
        begin
        form1.stgrid.Cells[0,((i) div (x_line_no))+1]:=floattostr(data[1,i]);
        end;
    for i:=0 to z_line_no-1 do
        begin
        form1.stgrid.Cells[((i) mod x_line_no)+1,((i) div
(x_line_no))+1]:=floattostr(data[2,i]);
        end;
    form1.stgrid.ColCount:=(x_line_no)+1;
    form1.stgrid.RowCount:=(y_line_no)+1;
    form1.stgrid.FixedCols:=1;
    form1.stgrid.Fixedrows:=1;
    x_spacing:=(strtofloat(form1.stgrid.Cells[2,0])-
strtofloat(form1.stgrid.Cells[1,0]));
    y_spacing:=(strtofloat(form1.stgrid.Cells[0,2])-
strtofloat(form1.stgrid.Cells[0,1]));
    setlength(data,0,0);
    setlength(data,x_line_no,y_line_no);
    for i:=0 to x_line_no-1 do
        begin
        for j:=0 to y_line_no-1 do
```

```
                begin
                data[i,j]:=strtofloat(form1.stgrid.Cells[i+1,j+1]);
                end;
            end;
        end;
        end;

procedure tdatagrid.read_noise;
        var
        f1:textfile;
        ch:char;
        ch1:shortstring;
        i:integer;
        j:integer;
        begin
        if form1.openxyzfile.Execute then
        begin
        setlength(data,0,0);
            for i:=0 to form1.stgrid.ColCount do
                for j:=0 to form1.stgrid.RowCount do
                begin
                form1.stgrid.Cells[i,j]:='';
                end;
            form1.stgrid.ColCount:=1;
            form1.stgrid.RowCount:=1;
            form1.stgrid.FixedCols:=0;
            form1.stgrid.FixedRows:=0;
            i:=0;
            assignfile(f1,form1.openxyzfile.FileName);
            reset(f1);
            while not eof(f1) do
                begin
                i:=i+1;
                readln(f1);
                end;
            setlength(data,3,i);
            z_line_no:=i;
            i:=0;
            j:=0;
            ch1:='';
            reset(f1);
            while not eof(f1) do
                begin
                read(f1,ch);
                if ((ch<>' ')and(ch<>#13))then
                    begin
                    if (ch<>#10)then
                        begin
                        ch1:=ch1+ch;
                        end;
```

```
                    if (ch=#10)then
                        begin
                        i:=i+1;
                        j:=0;
                        end;
                    end;
                if ((ch=' ')or(ch=#13))then
                    begin
                    data[j][i]:=strtofloat(ch1);
                    j:=j+1;
                    ch1:='';
                    end;
                end;
            closefile(f1);
            i:=0;
        while data[1,i]=data[1,0] do // # of line X lineno//
            begin
            i:=i+1;
            end;
        x_line_no:=i;
        y_line_no:=round(z_line_no/x_line_no);
        for i:=0 to z_line_no-1 do
            begin
            form1.stgrid.Cells[((i) mod x_line_no)+1,0]:=floattostr(data[0,i]);
            end;
        for i:=0 to z_line_no-1 do
            begin
            form1.stgrid.Cells[0,((i) div (x_line_no))+1]:=floattostr(data[1,i]);
            end;
        for i:=0 to z_line_no-1 do
            begin
            form1.stgrid.Cells[((i) mod x_line_no)+1,((i) div
(x_line_no))+1]:=floattostr(data[2,i]);
            end;
        form1.stgrid.ColCount:=(x_line_no)+1;
        form1.stgrid.RowCount:=(y_line_no)+1;
        form1.stgrid.FixedCols:=1;
        form1.stgrid.Fixedrows:=1;
        x_spacing:=(strtofloat(form1.stgrid.Cells[2,0])-
strtofloat(form1.stgrid.Cells[1,0]));
        y_spacing:=1;
        setlength(data,0,0);
        setlength(data,x_line_no,y_line_no);
        for i:=0 to x_line_no-1 do
            begin
            for j:=0 to y_line_no-1 do
                begin
                data[i,j]:=strtofloat(form1.stgrid.Cells[i+1,j+1]);
                end;
            end;
```

```
        end;
        end;

function tdatagrid.distance(x1,z1,x2,z2:double):double;
        var
        distance:double;
        begin
        distance:=sqrt(sqr(x1-x2)+sqr(z1-z2));
        result:=distance;
        end;
function tdatagrid.have_cross(x1,z1,x2,z2:double):boolean;
        begin
        if distance(x1,z1,x2,z2)>=lag then result:=true
        else result:=false;
        end;
procedure tdatagrid.find_limit(row_no:integer);
        var
        i:integer;
        begin
        i:=0;
        line_start_no:=0;
        line_end_no:=0;
        while (datagrid1.data[i,row_no]>(1.70141e37)) do
                begin
                i:=i+1;
                end;
        line_start_no:=i;
        i:=datagrid1.x_line_no-1;
        while (datagrid1.data[i,row_no]>(1.70141e37)) do
                begin
                i:=i-1;
                end;
        line_end_no:=i;
        end;

function tdatagrid.find_best_lag;
        var
        bestlag:double;
        j:integer;
        begin
        lag_zero:=0;
        for j:=0 to y_line_no-1 do
                begin
                find_limit(j);

bestlag:=distance(strtofloat(form1.stgrid.Cells[1,0])+line_start_no*x_spacing,data[lin
e_start_no,j],strtofloat(form1.stgrid.Cells[1,0])+line_end_no*x_spacing,data[line_end
_no,j]);
                if bestlag>lag_zero then lag_zero:=bestlag;
                end;
```

```
    end;

function tdatagrid.three_area(i1,j1,k1,i2,j2,k2,a5,b5,c5:double):double;
    var
    t1,t2,t3:double;
    sath:double;
    begin
    t1:=((j2-j1)*(c5-k1)-(k2-k1)*(b5-j1))*((j2-j1)*(c5-k1)-(k2-k1)*(b5-j1));
    t2:=((k2-k1)*(a5-i1)-(i2-i1)*(c5-k1))*((k2-k1)*(a5-i1)-(i2-i1)*(c5-k1));
    t3:=((i2-i1)*(b5-j1)-(j2-j1)*(a5-i1))*((i2-i1)*(b5-j1)-(j2-j1)*(a5-i1));
    sath:=((exp((0.5)*ln(t1+t2+t3)))/2);
    result:=sath;
    end;

function tdatagrid.sqr_area(a1,b1,c1,a2,b2,c2,a3,b3,c3,a4,b4,c4:double):double;
    var
    s1,s2,s3,s4:double;
    begin
    s1:=three_area(a1,b1,c1,a2,b2,c2,a5,b5,c5);
    s2:=three_area(a2,b2,c2,a3,b3,c3,a5,b5,c5);
    s3:=three_area(a3,b3,c3,a4,b4,c4,a5,b5,c5);
    s4:=three_area(a4,b4,c4,a1,b1,c1,a5,b5,c5);
    result:=(s1+s2+s3+s4);
    end;

function tdatagrid.total_area(length:integer):double;
    var
    i,j:integer;
    s:double;
    begin
    i:=0;
    j:=0;
    s:=0;
    while (j+length)<(y_line_no-1) do
        begin
        i:=0;
        while (i+length)<(x_line_no-1) do
            begin
            a1:=i*x_spacing;
            b1:=j*y_spacing;
            c1:=data[i,j];
            a2:=i*x_spacing;
            b2:=(j+length)*y_spacing;
            c2:=data[i,j+length];
            a3:=(i+length)*x_spacing;
            b3:=(j+length)*y_spacing;
            c3:=data[i+length,j+length];
            a4:=(i+length)*x_spacing;
            b4:=j*y_spacing;
            c4:=data[i+length,j];
```

```
                a5:=(i+(length/2))*x_spacing;
                b5:=(j+(length/2))*y_spacing;
                c5:=((c1+c2+c3+c4)/4);
                s:=s+(sqr_area(a1,b1,c1,a2,b2,c2,a3,b3,c3,a4,b4,c4));
                i:=i+length;
                end;
          j:=j+length;
          end;
    result:=s;
    end;

procedure tdatagrid.find_cross(x1,z1,x2,z2:double);
    begin
    x_start:=x1+lag*cos(arctan((z2-z1)/(x2-x1)));
    z_start:=z1+lag*sin(arctan((z2-z1)/(x2-x1)));
    end;

function tdatagrid.perimeter(row:integer):double;//row:satri az matrix data ke
mikhahim dimentione fractale anra bedast avarim;
    var
    i:integer;
    martabe:integer;
    begin
    martabe:=0;
    for i:=line_start_no to line_end_no-1 do
        begin
        x2:=(i+1)*x_spacing;
        z2:=data[i+1,row];
        while have_cross(x1,z1,x2,z2)=true do
            begin
            martabe:=martabe+1;
            find_cross(x1,z1,x2,z2);
            x1:=x_start;
            z1:=z_start;
            end;
        end;
    result:=(lag*(martabe+(distance(x1,z1,x2,z2))/lag)) ;
    end;

function gama1:double;
    var
    i,j:integer;
    sum,altitutesubtracting:double;
    k:integer;
    begin
    sum:=0;
    k:=0;
    for j:=0 to datagrid1.y_line_no-1 do
        begin
        datagrid1.find_limit(j);
```

```
            for i:=datagrid1.line_start_no to datagrid1.line_end_no-1 do
                begin
                altitutesubtracting:=(datagrid1.data[i,j]-datagrid1.data[i+1,j]);
                sum:=sum+(altitutesubtracting*altitutesubtracting);
                k:=k+1;
                end;
            end;
        result:=sum/(2*k);
        end;

function gama2:double;
        var
        i,j:integer;
        sum,altitutesubtracting:double;
        k:integer;
        begin
        sum:=0;
        k:=0;
        for j:=0 to datagrid1.y_line_no-1 do
            begin
            datagrid1.find_limit(j);
            for i:=datagrid1.line_start_no to datagrid1.line_end_no-2 do
                begin
                altitutesubtracting:=(datagrid1.data[i,j]-datagrid1.data[i+2,j]);
                sum:=sum+(altitutesubtracting*altitutesubtracting);
                k:=k+1;
                end;
            end;
        result:=sum/(2*k);
        end;

function log(x:double):double;
        begin
        result:=(ln(x)/2.3025850);
        end;

procedure TForm1.Open1Click(Sender: TObject);
begin
datagrid1.read_data;
form1.Caption:='Fractal method'+'  '+openxyzfile.FileName;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
datagrid1:=tdatagrid.create;
end;

procedure TForm1.floatingdividingscaleClick(Sender: TObject);
var
i:integer;
```

```
            lag1:=datagrid1.perimeter_matrix[0,j,k];
            end;
      datagrid1.perimeter_matrix_1[1,j]:=(perimeter);
      datagrid1.perimeter_matrix_1[0,j]:=(lag1);
      end;
form2.ShowModal;
end;

procedure TForm1.N2DfloatingdividingscaleClick(Sender: TObject);
var
i:integer;
begin
datagrid1.length:=1;
datagrid1.loop:=strtoint(inputbox('loop number','loop :','10'));
setlength(datagrid1.perimeter_matrix_1,2,datagrid1.loop);
for i:=0 to datagrid1.loop-1 do
      begin
      datagrid1.length:=i+1;

datagrid1.perimeter_matrix_1[0,i]:=(datagrid1.length)*(datagrid1.x_spacing)*(datagri
d1.y_spacing)*(datagrid1.length);
      datagrid1.perimeter_matrix_1[1,i]:=datagrid1.total_area(datagrid1.length);
      end;
end;

procedure TForm1.graph2dfdsClick(Sender: TObject);
begin
form2.ShowModal;
end;

procedure TForm1.Opennoise1Click(Sender: TObject);
begin
datagrid1.read_noise;
form1.Caption:='Fractal method'+'  '+openxyzfile.FileName;
end;

end.
```

پیوست دوم

کدهای اصلی فرم نمایش گراف نرم‌افزار

Fractal Methods
(FM)

```
unit Unit2;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, TeeProcs, TeEngine, Chart, Grids, Series, StdCtrls, Buttons;

type
  TForm2 = class(TForm)
    Chart1: TChart;
    Series1: TPointSeries;
    Series2: TLineSeries;
    sd1: TSaveDialog;
    Panel1: TPanel;
    BitBtn1: TBitBtn;
    stepbox: TCheckBox;
    BitBtn2: TBitBtn;
    Button1: TButton;
    Memo1: TMemo;
    procedure FormShow(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure Chart1Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }

  end;

var
  Form2: TForm2;

implementation

uses Unit1;
var
row_no:integer;
i:integer;
ln_matrix:array of array of double;
{$R *.DFM}

procedure TForm2.FormShow(Sender: TObject);
begin
row_no:=0;
i:=0;
chart1.Series[0].Clear;
end;
```

```
procedure TForm2.BitBtn1Click(Sender: TObject);
var
row:integer;
begin
if stepbox.Checked=true then
      begin
            if row_no<(datagrid1.y_line_no) then
                  begin
                  form2.Chart1.Series[0].Clear;
                  for row:=0 to loop_all-1 do
                        begin

form2.Chart1.Series[0].AddXY(datagrid1.perimeter_matrix[0,row,row_no],datagrid1.
perimeter_matrix[1,row,row_no],",clred);
                        end;
                  row_no:=row_no+1;
                  end;
      end;
end;

procedure TForm2.BitBtn2Click(Sender: TObject);
var
row:integer;
i:integer;
begin
for i:=0 to datagrid1.loop-1 do
      begin

form2.Chart1.Series[0].AddXY(datagrid1.perimeter_matrix_1[0,i],datagrid1.perimete
r_matrix_1[1,i],",clblue);
      end;
end;

procedure TForm2.Chart1Click(Sender: TObject);
begin
if form2.Chart1.Series[0].Marks.style=smslabel then
      begin
      form2.Chart1.Series[0].Marks.style:=smsxvalue;
      end
else
      begin
      form2.Chart1.Series[0].Marks.style:=smslabel;
      end;
end;

procedure TForm2.Button1Click(Sender: TObject);
begin
if sd1.Execute then
      begin
```

```
        form2.Chart1.SaveToBitmapFile(sd1.FileName);
        end;
end;

end.
```

پیوست سوم

کدهای اصلی فرم اصلی نرم‌افزار

Fractal Simulation
(FS)

```
unit Unit1;
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Menus, Grids, StdCtrls, ExtCtrls;
type tsimulate=class
private
public
delta_x:real;
data:array of array of real;
data_surf_matrix:array of array of real;
noise:array of real;
column:integer;
row:integer;
x_spacing,y_spacing:double;
x_line_no:integer;
y_line_no:integer;
line_start_no,line_end_no:integer;
z_line_no:integer;
dimention:double;
mean,std:real;
z_scale:real;
data_source:topendialog;
data_target:tsavedialog;
noise_target:tsavedialog;
procedure read_data;
procedure make_white_noise;
procedure process;
procedure save_data;
procedure save_noise;
procedure find_limit(row_no:integer);
function gama1:double;
function gama2:double;
constructor create;
end;


type
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    Exit1: TMenuItem;
    N2: TMenuItem;
    SaveAs1: TMenuItem;
    Open1: TMenuItem;
    run1: TMenuItem;
    Process1: TMenuItem;
    Help1: TMenuItem;
    openxyzfile: TOpenDialog;
```

```
    stgrid: TStringGrid;
    Panel1: TPanel;
    About1: TMenuItem;
    Savexyzfile: TSaveDialog;
    noisegrid: TStringGrid;
    N1: TMenuItem;
    ShowRandomNumberGrid1: TMenuItem;
    Graph1: TMenuItem;
    Savewhitenoise1: TMenuItem;
    savenoise: TSaveDialog;
    procedure FormCreate(Sender: TObject);
    procedure Open1Click(Sender: TObject);
    procedure Process1Click(Sender: TObject);
    procedure Graph1Click(Sender: TObject);
    procedure SaveAs1Click(Sender: TObject);
    procedure Savewhitenoise1Click(Sender: TObject);
  private
    { Private declarations }

  public
    { Public declarations }

  end;

var
  Form1: TForm1;

implementation

uses Unit2, Unit3;
{$R *.DFM}
function log(x:double):double;
      begin
      result:=(ln(x)/2.3025850);
      end;
function findmean(matrix:array of real;a:longint):real;
      var
      mean:real;
      sum:real;
      i:longint;
      begin
      sum:=0;
      for i:=0 to (a-1) do
            begin
            sum:=sum+matrix[i];
            end;
      mean:=sum/a;
      result:=mean;
      end;
function findstd(matrix:array of real;a:longint):real;
```

```
        var
        sum:real;
        mean:real;
        std:real;
        i:longint;
        begin
        mean:=findmean(matrix,a);
        sum:=0;
        for i:=0 to (a-1) do
              begin
              sum:=sum+(matrix[i]-mean)*(matrix[i]-mean);
              end;
        sum:=sum/(a-1);
        std:=sqrt(sum);
        result:=std;
        end;
function normalize(a:real;mean:real;std:real):real;
        var
        norm:real;
        begin
        norm:=((a-mean)/std);
        result:=norm;
        end;

procedure tsimulate.find_limit(row_no:integer);
        var
        i:integer;
        begin
        i:=0;
        line_start_no:=0;
        line_end_no:=0;
        while (data_surf_matrix[i,row_no]>(1.70141e37)) do
              begin
              i:=i+1;
              end;
        line_start_no:=i;
        i:=x_line_no-1;
        while (data_surf_matrix[i,row_no]>(1.70141e37)) do
              begin
              i:=i-1;
              end;
        line_end_no:=i;

        end;

function tsimulate.gama1:double;
        var
        i,j:integer;
        sum,altitutesubtracting:double;
        k:integer;
```

```
    begin
    sum:=0;
    k:=0;
    for j:=0 to y_line_no-1 do
        begin
        find_limit(j);
        for i:=line_start_no to line_end_no-1 do
            begin
            altitutesubtracting:=(data_surf_matrix[i,j]-data_surf_matrix[i+1,j]);
            sum:=sum+(altitutesubtracting*altitutesubtracting);
            k:=k+1;
            end;
        end;
    result:=sum/(2*k);
    end;

function tsimulate.gama2:double;
    var
    i,j:integer;
    sum,altitutesubtracting:double;
    k:integer;
    begin
    sum:=0;
    k:=0;
    for j:=0 to y_line_no-1 do
        begin
        find_limit(j);
        for i:=line_start_no to line_end_no-2 do
            begin
            altitutesubtracting:=(data_surf_matrix[i,j]-data_surf_matrix[i+2,j]);
            sum:=sum+(altitutesubtracting*altitutesubtracting);
            k:=k+1;
            end;
        end;
    result:=sum/(2*k);
    end;

procedure tsimulate.read_data;
    var
    i,j:integer;
    f1:textfile;
    ch:char;
    ch1:shortstring;
    h1,h2:double;
    begin
        begin
        SetLength(data,0,0);
        i:=0;
        assignfile(f1,data_source.FileName);
        reset(f1);
```

```
        while not eof(f1) do
              begin
              i:=i+1;
              readln(f1);
              end;
        setlength(data,3,i);
        z_line_no:=i;
        i:=0;
        j:=0;
        ch1:='';
        reset(f1);
        while not eof(f1) do
              begin
              read(f1,ch);
              if ((ch<>' ')and(ch<>#13))then
                    begin
                    if (ch<>#10)then
                          begin
                          ch1:=ch1+ch;
                          end;
                    if (ch=#10)then
                          begin
                          i:=i+1;
                          j:=0;
                          end;
                    end;
              if ((ch=' ')or(ch=#13))then
                    begin
                    data[j][i]:=strtofloat(ch1);
                    j:=j+1;
                    ch1:='';
                    end;
              end;
        closefile(f1);
        i:=0;
while data[1,i]=data[1,0] do
        begin
        i:=i+1;
        end;
x_line_no:=i;
y_line_no:=round(z_line_no/x_line_no);
setlength(data_surf_matrix,x_line_no+1,y_line_no+1);
for i:=0 to z_line_no-1 do
        begin
        data_surf_matrix[((i) mod x_line_no)+1,0]:=(data[0,i]);
        end;

for i:=0 to z_line_no-1 do
        begin
        data_surf_matrix[0,((i) div (x_line_no))+1]:=(data[1,i]);
```

```
            begin
            for i:=1 to 2*x_line_no-1 do
                  begin
                  data[2][k]:=(data_surf_matrix[i,j]);
                  k:=k+1;
                  end;
            end;
      assignfile(f,data_target.FileName);
      rewrite(f);
      for i:=0 to matrixrow do
            begin
            ch:=(floattostr(data[0][i])+' '+floattostr(data[1][i])+' '+floattostr(data[2][i]));
            writeln(f,ch);
            end;
      closefile(f);
      end;
end;
procedure tsimulate.save_noise;
var
matrixrow:integer;
ch:shortstring;
F: TextFile;
i,j,k:integer;
begin
if noise_target.Execute then
      begin
      k:=0;
      setlength(data,3,(2*x_line_no-1)*(2*y_line_no-1));
      matrixrow:=(2*x_line_no-1)*(2*y_line_no-1)-1;
      for i:=0 to matrixrow do
            begin
            data[0][i]:=i;
            data[1][i]:=0;
            end;
      for j:=0 to (2*x_line_no-1)*(2*y_line_no-1)-1 do
            begin
            data[2][j]:=noise[j]
            end;
      assignfile(f,noise_target.FileName);
      rewrite(f);
      for i:=0 to matrixrow do
            begin
            ch:=(floattostr(data[0][i])+' '+floattostr(data[1][i])+' '+floattostr(data[2][i]));
            writeln(f,ch);
            end;
      closefile(f);
      end;
end;
procedure tsimulate.make_white_noise;
      var
```

```
        i:integer;
        begin
        setlength(noise,(2*x_line_no-1)*(2*y_line_no-1));
        randomize;
        for i:=0 to (2*x_line_no-1)*(2*y_line_no-1)-1 do
                begin
                noise[i]:=random;
                end;
        mean:=findmean(noise,(2*x_line_no-1)*(2*y_line_no-1));
        std:=findstd(noise,(2*x_line_no-1)*(2*y_line_no-1));
        for i:=0 to (2*x_line_no-1)*(2*y_line_no-1)-1 do
                begin
                noise[i]:=normalize(noise[i],mean,std);
                end;
        end;
procedure tsimulate.process;
        var
        count:longint;
        i,j,mcount:integer;
        m1,m2,m3,m4:real;
        mid:real;
        begin
        count:=0;
        z_scale:=strtofloat(inputbox('Random Number Generator Scale','Scale :','1'));
        showmessage('The White Random Number Generater is OK '+'  '+'Mean =
'+floattostr(findmean(noise,(2*x_line_no-1)*(2*y_line_no-1)))+'  '+'Variance =
'+floattostr(findstd(noise,(2*x_line_no-1)*(2*y_line_no-1))));
        application.ProcessMessages;
        setlength(data_surf_matrix,2*(x_line_no),2*(y_line_no));
        for i:=1 to (2*(x_line_no)-1) do
                begin
                if (i mod 2)=0 then data_surf_matrix[i,0]:=(((data_surf_matrix[i-
1,0])+(data_surf_matrix[i+1,0]))/2);
                end;
        for i:=1 to (2*(y_line_no)-1) do
                begin
                if (i mod 2)=0 then data_surf_matrix[0,i]:=(((data_surf_matrix[0,i-
1])+(data_surf_matrix[0,i+1]))/2);
                end;
        for i:=1 to (2*(x_line_no)-1) do
                begin
                for j:=1 to (2*(y_line_no)-1) do
                        begin
                        if (((i mod 2)=0) and ((j mod 2)=0)) then
                                begin
                                m1:=0;
                                m2:=0;
                                m3:=0;
                                m4:=0;
                                mcount:=0;
```

```
                    if data_surf_matrix[i+1,j+1]<1.70141e38 then
                        begin
                        m1:=data_surf_matrix[i+1,j+1];
                        mcount:=mcount+1;
                        end;
                    if data_surf_matrix[i-1,j-1]<1.70141e38 then
                        begin
                        m2:=data_surf_matrix[i-1,j-1];
                        mcount:=mcount+1;
                        end;
                    if data_surf_matrix[i+1,j-1]<1.70141e38 then
                        begin
                        m3:=data_surf_matrix[i+1,j-1];
                        mcount:=mcount+1;
                        end;
                    if data_surf_matrix[i-1,j+1]<1.70141e38 then
                        begin
                        m4:=data_surf_matrix[i-1,j+1];
                        mcount:=mcount+1;
                        end;
                    mid:=m1+m2+m3+m4;
                    if mcount>1 then
                        begin
                        data_surf_matrix[i,j]:=(z_scale*noise[count]+(mid/mcount));
                        end
                        else
                        begin
                        data_surf_matrix[i,j]:=1.70141e38;
                        end;
                    count:=count+1;
                    end;
                end;
            end;

    for i:=2 to (2*(x_line_no)-2) do
            begin
            for j:=2 to (2*(y_line_no)-2) do
                begin
                if ( (((i mod 2)=0) and ((j mod 2)<>0)) or (((i mod 2)<>0) and ((j mod
2)=0)) ) then
                    begin
                    m1:=0;
                    m2:=0;
                    m3:=0;
                    m4:=0;
                    mcount:=0;
                    if data_surf_matrix[i,j+1]<1.70141e38 then
                        begin
                        m1:=data_surf_matrix[i,j+1];
                        mcount:=mcount+1;
```

```
                    end;
                if data_surf_matrix[i,j-1]<1.70141e38 then
                    begin
                    m2:=data_surf_matrix[i,j-1];
                    mcount:=mcount+1;
                    end;
                if data_surf_matrix[i+1,j]<1.70141e38 then
                    begin
                    m3:=data_surf_matrix[i+1,j];
                    mcount:=mcount+1;
                    end;
                if data_surf_matrix[i-1,j]<1.70141e38 then
                    begin
                    m4:=data_surf_matrix[i-1,j];
                    mcount:=mcount+1;
                    end;
                mid:=m1+m2+m3+m4;
                if mcount>1 then
                    begin

data_surf_matrix[i,j]:=(z_scale*noise[count]/1.41421356+(mid/mcount));
                    end
                    else
                    begin
                    data_surf_matrix[i,j]:=1.70141e38;
                    end;
                count:=count+1;
                end;
            end;
        end;

    for i:=1 to (2*(x_line_no)-1) do
        begin
        if (i mod 2)=0 then
            begin
            m1:=0;
            m2:=0;
            m3:=0;
            mcount:=0;
            if data_surf_matrix[i-1,1]<1.70141e38 then
                begin
                m1:=data_surf_matrix[i-1,1];
                mcount:=mcount+1;
                end;
            if data_surf_matrix[i+1,1]<1.70141e38 then
                begin
                m2:=data_surf_matrix[i+1,1];
                mcount:=mcount+1;
                end;
            if data_surf_matrix[i,2]<1.70141e38 then
```

```
                    begin
                    m3:=data_surf_matrix[i,2];
                    mcount:=mcount+1;
                    end;
                mid:=m1+m2+m3;
                if mcount>1 then
                        begin

data_surf_matrix[i,1]:=(z_scale*noise[count]/1.41421356+(mid/mcount));
                        end
                        else
                        begin
                        data_surf_matrix[i,1]:=1.70141e38;
                        end;
                count:=count+1;
                m1:=0;
                m2:=0;
                m3:=0;
                mcount:=0;
                if data_surf_matrix[i-1,2*(y_line_no)-1]<1.70141e38 then
                    begin
                    m1:=data_surf_matrix[i-1,2*(y_line_no)-1];
                    mcount:=mcount+1;
                    end;
                if data_surf_matrix[i+1,2*(y_line_no)-1]<1.70141e38 then
                    begin
                    m2:=data_surf_matrix[i+1,2*(y_line_no)-1];
                    mcount:=mcount+1;
                    end;
                if data_surf_matrix[i,2*(y_line_no)-2]<1.70141e38 then
                    begin
                    m3:=data_surf_matrix[i,2*(y_line_no)-2];
                    mcount:=mcount+1;
                    end;
                mid:=m1+m2+m3;
                if mcount>1 then
                        begin
                        data_surf_matrix[i,2*(y_line_no)-
1]:=(z_scale*noise[count]/1.41421356+(mid/mcount));
                        end
                        else
                        begin
                        data_surf_matrix[i,2*(y_line_no)-1]:=1.70141e38;
                        end;
                count:=count+1;
                end;
        end;

    for j:=1 to (2*(y_line_no)-1) do
        begin
```

```
        if (j mod 2)=0 then
              begin
              m1:=0;
              m2:=0;
              m3:=0;
              mcount:=0;
              if data_surf_matrix[1,j-1]<1.70141e38 then
                   begin
                   m1:=data_surf_matrix[1,j-1];
                   mcount:=mcount+1;
                   end;
              if data_surf_matrix[1,j+1]<1.70141e38 then
                   begin
                   m2:=data_surf_matrix[1,j+1];
                   mcount:=mcount+1;
                   end;
              if data_surf_matrix[2,j]<1.70141e38 then
                   begin
                   m3:=data_surf_matrix[2,j];
                   mcount:=mcount+1;
                   end;
              mid:=m1+m2+m3;
              if mcount>1 then
                        begin

data_surf_matrix[1,j]:=(z_scale*noise[count]/1.41421356+(mid/mcount));
                        end
                        else
                        begin
                        data_surf_matrix[1,j]:=1.70141e38;
                        end;
              count:=count+1;
              m1:=0;
              m2:=0;
              m3:=0;
              mcount:=0;
              if data_surf_matrix[2*(x_line_no)-1,j-1]<1.70141e38 then
                   begin
                   m1:=data_surf_matrix[2*(x_line_no)-1,j-1];
                   mcount:=mcount+1;
                   end;
              if data_surf_matrix[2*(x_line_no)-1,j+1]<1.70141e38 then
                   begin
                   m2:=data_surf_matrix[2*(x_line_no)-1,j+1];
                   mcount:=mcount+1;
                   end;
              if data_surf_matrix[2*(x_line_no)-2,j]<1.70141e38 then
                   begin
                   m3:=data_surf_matrix[2*(x_line_no)-2,j];
                   mcount:=mcount+1;
```

```
                    end;
                mid:=m1+m2+m3;
                if mcount>1 then
                        begin
                        data_surf_matrix[2*(x_line_no)-
1,j]:=(z_scale*noise[count]/1.41421356+(mid/mcount));
                        end
                        else
                        begin
                        data_surf_matrix[2*(x_line_no)-1,j]:=1.70141e38;
                        end;
                count:=count+1;
                end;
            end;
        showmessage('# of lines (X direction) = '+inttostr(2*(x_line_no)-1)+'     # of
lines (y direction) = '+inttostr(2*(y_line_no)-1));
        end;

constructor tsimulate.create;
        begin
        setlength(data,0,0);
        setlength(data_surf_matrix,0,0);
        delta_x:=1;
        mean:=0;
        std:=1;
        setlength(noise,0);
        row:=0;
        column:=0;
        x_line_no:=1;
        y_line_no:=1;
        z_line_no:=1;
        z_scale:=1;
        end;

var
sim1:tsimulate;
procedure TForm1.FormCreate(Sender: TObject);
begin
sim1:=tsimulate.create;
end;

procedure TForm1.Open1Click(Sender: TObject);
var
i,j:integer;
begin
sim1:=tsimulate.create;
sim1.data_source:=form1.openxyzfile;
sim1.data_target:=form1.Savexyzfile;
sim1.noise_target:=form1.savenoise;
if openxyzfile.Execute then
```

```
begin
sim1.read_data;
sim1.make_white_noise;
form1.stgrid.ColCount:=2*sim1.x_line_no;
form1.stgrid.RowCount:=2*sim1.y_line_no;
form1.stgrid.FixedCols:=1;
form1.stgrid.FixedRows:=1;
for i:=0 to (2*(sim1.x_line_no))-1 do
      begin
      for j:=0 to (2*(sim1.y_line_no))-1 do
            begin
            form1.stgrid.Cells[i,j]:=floattostr(sim1.data_surf_matrix[i,j]);
            end;
      end;
form1.Caption:='Fractal simulation  '+sim1.data_source.FileName;
end;
end;
procedure TForm1.Process1Click(Sender: TObject);
var
i,j:integer;
h1,h2:double;
oldx_line_no,oldy_line_no:integer;
begin
sim1.process;
for i:=0 to (2*(sim1.x_line_no))-1 do
      begin
      for j:=0 to (2*(sim1.y_line_no))-1 do
            begin
            form1.stgrid.Cells[i,j]:=floattostr(sim1.data_surf_matrix[i,j]);
            end;
      end;
setlength(sim1.data_surf_matrix,0,0);
setlength(sim1.data_surf_matrix,2*sim1.x_line_no-1,2*sim1.y_line_no-1);
for i:=0 to 2*sim1.x_line_no-1-1 do
      begin
      for j:=0 to 2*sim1.y_line_no-1-1 do
            begin
            sim1.data_surf_matrix[i,j]:=strtofloat(form1.stgrid.Cells[i+1,j+1]);
            end;
      end;
oldx_line_no:=sim1.x_line_no;
oldy_line_no:=sim1.y_line_no;
sim1.x_line_no:=2*sim1.x_line_no-1;
sim1.y_line_no:=2*sim1.y_line_no-1;
sim1.x_spacing:=0;
sim1.y_spacing:=0;
h1:=0;
h2:=0;
sim1.dimention:=0;
```

فهرست منابع به ترتیب
استفاده در متن

[۱] امینی نجفی، " با هندسه فراکتال آشنا شویم"، مجله رشد آموزش زمین‌شناسی، بهار و تابستان ۱۳۷۲

[۲] تخم چی ـ بهزاد، " شبیه‌سازی و مدلسازی زمین آماری و هندسه فراکتال"، سمینار کارشناسی ارشد، دانشگاه صنعتی امیرکبیر، تابستان ۱۳۷۵

[3] Barabasi, A.L. and Stanley, H.E., "Fractal concepts in surface growth" Cambridge University Press, 1995

[۴] شهریاری ـ سهراب، خطیب ـ محمد مهدی، " تحلیل فراکتالی سیستم گسل نهبندان"، مجله علوم زمین، شماره ۲۳ و ۲۴، بهار و تابستان ۷۶، صفحه ۲۲ تا ۳۹

[۵] ذوالفقاری، محمود، " نقشه‌برداری"، چاپ دهم، خرداد ۱۳۷۰

[6] Kuchta, Marke. "Improved mine planning using geostatistical and Fractal geometry for geologic modeling", Lulea, Sweden – June 1990

[۷] مدنی ـ حسن، " مبانی زمین‌آمار " ، انتشارات دانشگاه صنعتی امیرکبیر ـ واحد تفرش، ۱۳۷۳

[8] Journal, A.C and Huijbregts "Mining geostatistics" Academic Press - 1990