LAPORAN
CLEAN MESSY DATA AND LOCATE EXTREMA USING LIVE EDITOR TASKS

Disusun oleh :

Nama :

NPM

Dosen :

Joko Triloka, S.Kom., M.T., Ph.D

PROGRAM STUDI MAGISTER TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
INSTITUT INFORMATIKA DAN BISNIS DARMAJAYA
TAHUN 2024

KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Dengan mengucapkan Alhamdulillah segala puji dan syukur penulis panjatkan atas kehadirat Allah SWT, karena berkat rahmat dan hidayah-Nya penulis bisa menyelesaikan Laporan yang berisi tentang **CLEAN MESSY DATA AND LOCATE EXTREMA USING LIVE EDITOR TASKS** yang terdapat Matlab Live Editor dengan alamat website https://www.mathworks.com/help/matlab/data_analysis/cleandatawithliveeditortasks.html.
Laporan menjelaskan tentang komponen-komponen yang ada pada Create Plot, Fill Missing data, Fill Outliers, Smooth Data dan Locate Extrema. Laporan juga dilengkapi dengan Uji Coba dengan merubah komponen-komponen yang ada pada Create Plot, Fill Missing data, Fill Outliers, Smooth Data dan Locate Extrema.
Untuk Mempermudah membaca laporan, berikut nomor halaman yang sudah disediakan :

Penulis berharap semoga laporan ini dapat bermanfaat bagi pembaca dan dapat dijadikan referensi demi pengembangan ke arah yang lebih baik. Kebenaran datangnya dari Allah SWT dan kesalahan datangnya dari diri penulis. Semoga Allah SWT senantiasa melimpahkan Rahmat dan Ridho-Nya kepada kita semua.

Wassalamu' alaikum Wr. Wb.

Bandar Lampung, Oktober 2024
Penulis,

A. CREATE PLOT

Hal pertama yang harus dilakukan yaitu membuat dan memplot data vektor yang berantakan. Data tersebut mengandung empat nilai *NaN* (nilai hilang) dan lima *outlier* (nilai ekstrem). Kode yang digunakan:

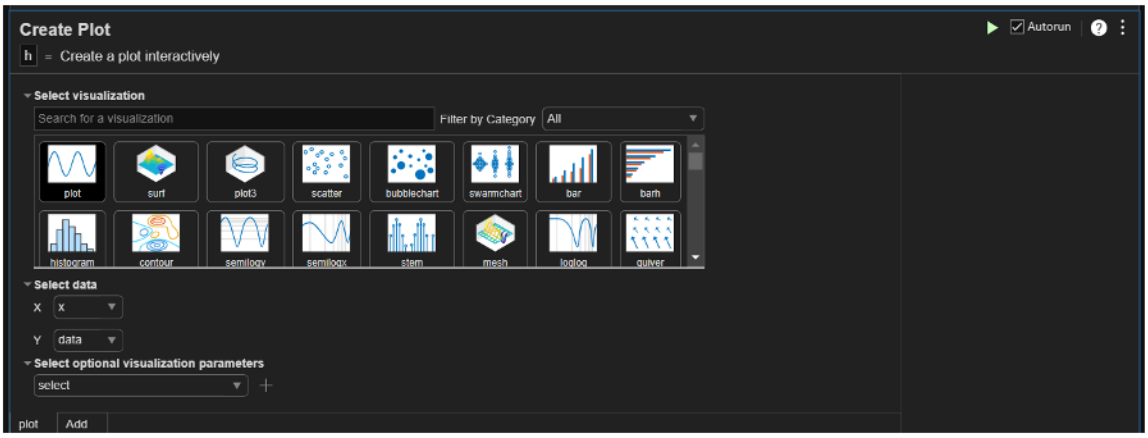
```
x = 1:100;
data = cos(2*pi*0.05*x + 2*pi*rand) + 0.5*randn(1,100);
data(20:20:80) = NaN;
data(10:20:90) = [-50 40 30 -45 35];
```

Kode tersebut membuat data vektor yang memiliki 100 nilai (dari 1 hingga 100). Fungsi cos() digunakan untuk membentuk pola sinusoidal yang ditambah dengan sedikit noise acak menggunakan randn(). Namun, beberapa nilai pada data sengaja diganti dengan nilai NaN (hilang) dan outlier ekstrem (-50, 40, 30, -45, 35) untuk mensimulasikan data yang berantakan. 2*pi*0.05*x memberi frekuensi gelombang, sementara 2*pi*rand memberikan pergeseran fase acak. Kemudian, randn() menambah *noise* acak ke data.

Dalam kode data(20:20:80) = NaN;, ini berarti kamu mengganti data pada posisi **kelipatan 20** (yaitu: 20, 40, 60, 80) menjadi *NaN*. Jadi, ada empat posisi yang diganti, bukan hanya satu. Sehingga data di posisi 20, 40, 60, dan 80 semua menjadi *NaN*, menandakan nilai hilang pada data tersebut.

Kode data(10:20:90) = [-50 40 30 -45 35];, Nilai dari data(10:20:90), 20 disini untuk mengambil data dalam rentang tertentu dengan langkah 20 sedangkan 10 untuk awal indeks dimulai dan untuk 90 untuk akhir indeks sehingga didapat nilai data pada indeks ke 10, 30, 50, 70, dan 90. Kemudian nilai tersebut diganti dengan nilai tertentu: -50, 40, 30, -45, dan 35. Ini bertujuan untuk mensimulasikan *outlier*, yaitu nilai-nilai ekstrem yang jauh dari pola data yang normal. Jadi, angka-angka ini membuat data menjadi tidak normal dan memperlihatkan adanya gangguan pada data tersebut.

Setelah data plot sudah dibuat, Matlab menyediakan visualisasi gambar plot yang beragam dan sesuai keinginan pengguna, berikut tampilan untuk memilih visualisasi plot :



Berikut penjelesan dari komponen-komponen yang terdapat pada Create plot seperti gambar diatas:

- **Select Visualization**

  Berikut adalah penjelasan tentang beberapa jenis visualisasi yang tersedia pada alat yang terlihat pada gambar :

1. Plot : Visualisasi standar yang menghubungkan titik data dengan garis. Ini berguna untuk menunjukkan tren data atau hubungan antar variabel.
2. Surf : Membuat grafik permukaan 3D. Cocok untuk memvisualisasikan hubungan antara tiga variabel dan bagaimana mereka berubah di seluruh ruang tiga dimensi.
3. Plot3 : Grafik 3D standar yang digunakan untuk menggambarkan hubungan antara tiga variabel dengan menggunakan titik-titik di ruang tiga dimensi.
4. Scatter : Menampilkan data dalam bentuk titik yang tersebar pada bidang 2D. Scatter plot sering digunakan untuk mengidentifikasi hubungan atau pola antara dua variabel.
5. Bubblechart : Mirip dengan scatter plot, namun setiap titik data direpresentasikan sebagai gelembung, di mana ukuran gelembung mewakili variabel ketiga.
6. Swarmchart : Memvisualisasikan distribusi data dengan menempatkan titik-titik pada baris horizontal tanpa tumpang tindih. Cocok untuk melihat kepadatan data.
7. Bar : Diagram batang yang digunakan untuk memvisualisasikan perbandingan antar kategori. Panjang batang merepresentasikan nilai dari setiap kategori.
8. Barh : Sama seperti diagram batang (bar), namun sumbu X dan Y ditukar, sehingga batang ditampilkan secara horizontal.

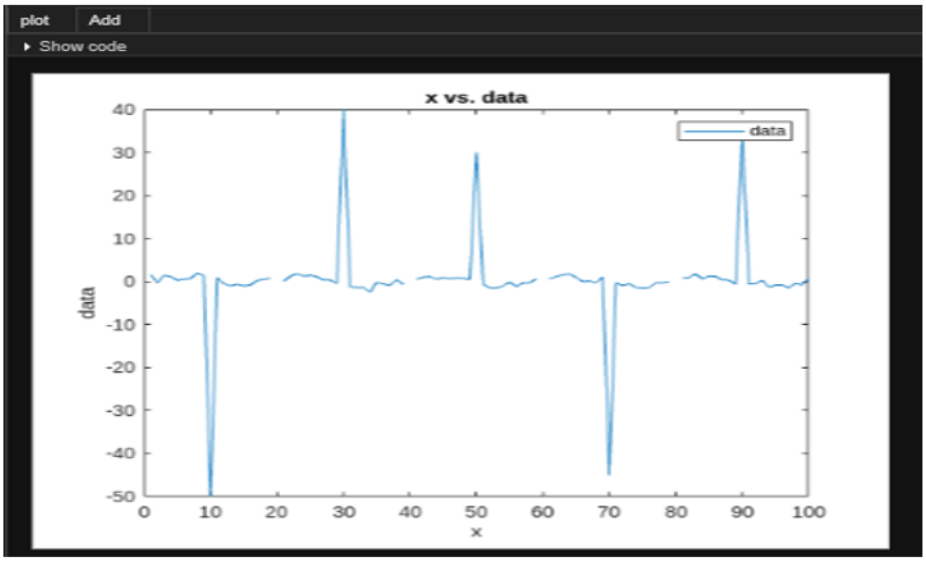
9. Histogram : Menampilkan distribusi data numerik dengan membagi data ke dalam interval atau bin, kemudian menampilkan frekuensi data dalam setiap bin. Histogram sering digunakan untuk menganalisis distribusi frekuensi.
10. Contour : Memvisualisasikan data 3D dalam bentuk peta kontur 2D. Garis kontur menghubungkan titik dengan nilai yang sama, sering digunakan untuk visualisasi peta topografi atau distribusi.
11. Semilog : Plot yang menggunakan skala logaritmik pada satu sumbu (biasanya sumbu Y). Digunakan saat rentang data sangat besar atau data bersifat eksponensial.
12. Semilogx : Sama seperti semilog, namun skala logaritmik diterapkan pada sumbu X.
13. Stem : Mirip dengan plot garis, tetapi data direpresentasikan dengan batang vertikal dari sumbu X menuju titik data, lebih jelas untuk melihat setiap titik data.
14. Mesh : Membuat grafik wireframe (kerangka kawat) dalam 3D. Biasanya digunakan untuk melihat data permukaan.
15. Lologlog : Plot dengan kedua sumbu menggunakan skala logaritmik. Berguna saat rentang nilai untuk kedua variabel sangat luas.
16. Quiver : Grafik vektor yang menunjukkan arah dan kecepatan aliran, sering digunakan dalam visualisasi medan angin atau aliran fluida.

Diatas merupakan penjelasan dari beberapa jenis visualisasi yang ada digambar tersebut, sebenarnya ada banyak jenis visualisasi di gambar tersebut, hanya diambil beberapa contoh penjelasan saja.

- Bagian **"Filter by Category"** dalam konteks visualisasi seperti yang terlihat pada gambar memungkinkan Anda untuk menyaring jenis-jenis visualisasi berdasarkan kategori tertentu, agar lebih mudah menemukan visualisasi yang sesuai dengan kebutuhan. Ini berguna jika alat ini memiliki banyak pilihan jenis grafik dan ingin membatasi atau mencari tipe visualisasi spesifik.
- Pada bagian **“Select Data"**, bisa memilih data yang ingin divisualisasikan pada grafik.
  - X : Anda akan memilih kolom atau variabel yang akan dipetakan pada sumbu X. Ini adalah variabel independen yang akan menggerakkan grafik.
  - Y : Anda memilih kolom atau variabel yang akan dipetakan pada sumbu Y. Ini adalah variabel dependen, yang nilainya dipengaruhi oleh sumbu X.

- Untuk bagian **"Select Optional Visualization Parameters"**, memungkinkan untuk menyesuaikan visualisasi lebih lanjut dengan mengubah parameter opsional untuk tampilan grafik.

Berikut hasil gambar visualisasi dari data plot yang dibuat :



Berikut penjelasan dari gambar diatas :

1. Sumbu X dan Sumbu Y:

- Sumbu X adalah rentang dari 1 hingga 100, yang dihasilkan oleh x = 1:100;.
- Sumbu Y (data) dihasilkan oleh kombinasi fungsi cosinus, noise acak, dan beberapa titik yang dimodifikasi secara eksplisit seperti yang didefinisikan dalam kode.

2. Gelombang Sinusoidal dengan Noise:

- Baris kode ini:

  data = cos(2*pi*0.05*x + 2*pi*rand) + 0.5*randn(1,100);

menghasilkan **gelombang sinusoidal** dengan sedikit gangguan acak (noise). Ini terlihat pada sebagian besar grafik, terutama di antara titik-titik di mana tidak ada modifikasi eksplisit. Berikut penjelasannya :

- Variabel data dihasilkan dari kombinasi fungsi sinusoidal cosinus dengan beberapa tambahan noise acak.
- cos(2*pi*0.05*x) menghasilkan gelombang sinusoidal yang periodik dengan frekuensi rendah (periode yang lebih panjang).

- 2*pi*rand menambahkan offset acak ke fungsi cosinus tersebut, sehingga sinyal tidak mulai dari posisi tetap.
- 0.5*randn(1,100) menambahkan noise Gaussian (acak) dengan distribusi normal pada sinyal tersebut. Faktor 0.5 mengurangi amplitudo noise.

3. Nilai NaN (Data Hilang):

- Baris ini:

  data(20:20:80) = NaN;

menggantikan data pada indeks 20, 40, 60, dan 80 dengan NaN, yang menyebabkan tidak ada nilai yang diplot pada titik-titik tersebut. Ini menghasilkan celah yang terlihat pada plot, di mana tidak ada garis atau titik di posisi tersebut.

4. Nilai Ekstrem di Titik Tertentu:

- Baris ini:

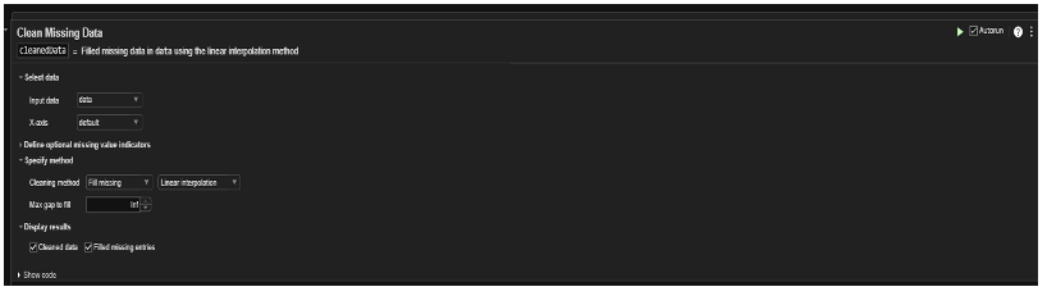
  data(10:20:90) = [-50 40 30 -45 35];

mengganti nilai pada indeks ke-10, 30, 50, 70, dan 90 dengan nilai spesifik: -50, 40, 30, -45, dan 35. Ini menjelaskan mengapa ada puncak tajam yang terlihat pada plot, terutama di titik-titik tersebut, dengan nilai yang sangat jauh berbeda dari gelombang sinusoidal normal. Pada plot:

- **X = 10**: Lembah dalam (nilai -50).
- **X = 30**: Puncak tinggi (nilai 40).
- **X = 50**: Puncak tinggi lagi (nilai 30).
- **X = 70**: Lembah dalam lagi (nilai -45).
- **X = 90**: Puncak tinggi lagi (nilai 35).

B. FILL MISSING DATA

Bagian ini menjelaskan secara umum bahwa alat ini digunakan untuk mengganti nilai NaN yang ada di dataset dengan metode interpolasi atau metode lain. Hasilnya dapat divisualisasikan setelah data yang hilang diisi.



Clean Missing Data

Ini adalah tugas yang akan lakukan, yaitu membersihkan data yang hilang (NaN). Di sini terdapat komponen penting yaitu cleaneddata merupakan variabel atau nama dari hasil dataset baru yang sudah diisi.

Berikut penjelasan dari komponen-komponen yang ada pada Clean Missing Data pada gambar diatas :

- Pada bagian **Select Data** terdapat dua input yaitu :
  - Input Data : Di bagian ini, bisa memilih dataset mana yang akan diproses untuk mengisi nilai NaN. Contoh pada gambar, telah memilih variabel data sebagai input. Variabel data adalah dataset yang akan dibersihkan dari nilai NaN.
  - X-axis : Pilihan ini menunjukkan sumbu X yang digunakan dalam data tersebut. Biasanya, sumbu X bisa menggunakan default jika tidak ada pengaturan khusus, atau bisa dipilih variabel lain jika ingin mengatur ulang.
- Pada bagian **Define Optional Missing Value** Indicators, bisa menentukan indikator khusus untuk nilai hilang (misalnya, simbol atau nilai tertentu yang menandakan data hilang) jika diperlukan. Ini adalah opsional, jadi Anda tidak wajib mengubahnya jika indikator NaN sudah cukup.
- Pada bagian **Specify Method** merupakan jantung dari proses pengisian data hilang:
  - Cleaning Method: Bisa memilih cara untuk membersihkan data yang hilang. Ada tiga pilihan Fill missing, remove missing atau none. Berikut penjelasan dari beberapa metode yang ada :
    1. Constant Value : Metode ini mengisi nilai yang hilang (NaN) dengan nilai konstan yang ditentukan. Misalnya, bisa mengganti semua NaN dengan angka 0 atau angka tertentu.
    2. Previous Value : Mengisi nilai yang hilang dengan nilai sebelumnya (yang muncul sebelum NaN). Metode ini sering digunakan dalam data deret waktu di mana pola data dari waktu ke waktu cenderung berkelanjutan.
    3. Next Value : Kebalikan dari Previous Value, metode ini mengisi nilai NaN dengan nilai yang muncul setelahnya. Biasanya digunakan jika data secara logis cenderung memprediksi nilai saat ini dari masa depan.

4. Nearest Value : Mengganti nilai NaN dengan nilai terdekat di dataset, baik nilai sebelumnya maupun nilai setelahnya. Ini akan memilih nilai yang memiliki jarak terkecil ke nilai yang hilang.
5. Linear Interpolation : Mengisi nilai yang hilang dengan interpolasi linier, yaitu menghitung nilai NaN berdasarkan dua titik tetangga terdekatnya menggunakan garis lurus. Ini adalah metode yang sangat umum dan efektif untuk mengisi data kontinu.
6. Spline Interpolation : Mengisi nilai yang hilang dengan interpolasi spline, yaitu menggunakan fungsi spline untuk menghitung nilai di antara titik data yang ada. Interpolasi spline lebih halus dibandingkan interpolasi linier karena menggunakan polinomial yang menghubungkan titik-titik data.
7. Shape-preserving cubic interpolation (PCHIP) : Menggunakan interpolasi kubik yang menjaga bentuk data, sehingga interpolasi ini tidak menghasilkan osilasi yang berlebihan (over-shooting) di antara titik-titik. Metode ini sering digunakan ketika ingin menjaga kehalusan kurva data tetapi tetap realistis.
8. Modified Akima cubic interpolation : Seperti interpolasi kubik, tetapi menggunakan metode Akima yang cenderung lebih stabil dan menjaga kontinuitas data. Ini digunakan untuk data yang memiliki fluktuasi yang lebih ekstrim.
9. Moving Median : Metode ini mengisi nilai yang hilang dengan median dari beberapa nilai di sekitar NaN. Ini membantu mengurangi pengaruh outlier, cocok untuk data yang memiliki banyak fluktuasi.
10. Moving Mean : Mengisi nilai yang hilang dengan rata-rata (mean) dari beberapa nilai di sekitar NaN. Ini lebih sensitif terhadap outlier dibandingkan median, tetapi memberikan hasil yang halus untuk data yang stabil.
11. Custom Function : Bisa mendefinisikan fungsi sendiri untuk mengisi nilai yang hilang. Ini memungkinkan fleksibilitas yang lebih besar karena bisa menentukan logika yang tepat untuk memperlakukan data NaN sesuai kebutuhan.

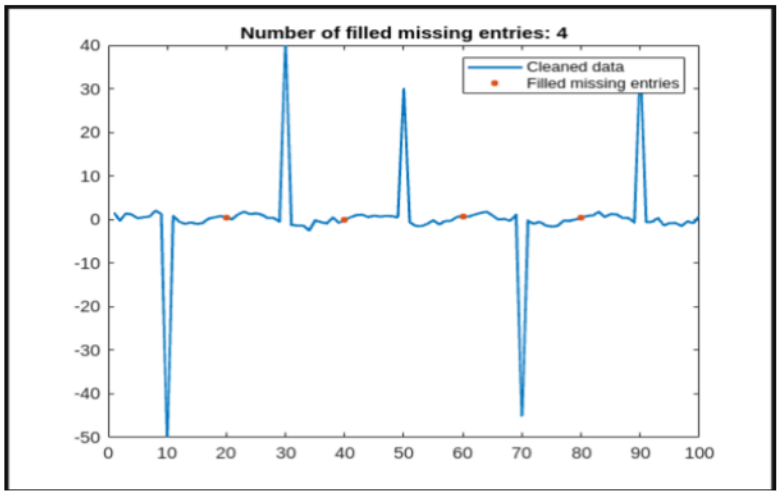
- Max Gap to Fill : Ini adalah jumlah maksimum celah atau nilai yang hilang berturut-turut yang dapat diisi.

- Pada **Bagian Display Results** ada dua bagian yang bisa di pilih untuk ditampilkan :
  - ✓ Cleaned Data : Menandakan bahwa hasil data yang sudah dibersihkan akan ditampilkan.

- ✓ Filled Missing Entries : Menandakan bahwa bagian yang sudah diisi dengan nilai interpolasi akan ditampilkan, sehingga bisa melihat nilai mana yang sebelumnya hilang dan kini telah diisi.

➢ Pada Bagian **Show Code** memungkinkan untuk melihat kode yang dihasilkan oleh alat ini setelah menjalankan proses pengisian data. Fitur ini berguna jika ingin mengetahui secara teknis apa yang terjadi di balik layar dan memungkinkan untuk mereplikasi atau memodifikasi kodenya.

Berikut hasil gambar setelah data yang hilang sudah di isi sesuai dengan input gambar sebelumnya di Fill Missing Data :



Gambar grafik hasil dari proses fill missing data. Grafik ini memperlihatkan data yang telah di-cleaned dan beberapa titik di mana data yang hilang telah diisi. Penjelasan lebih rinci:

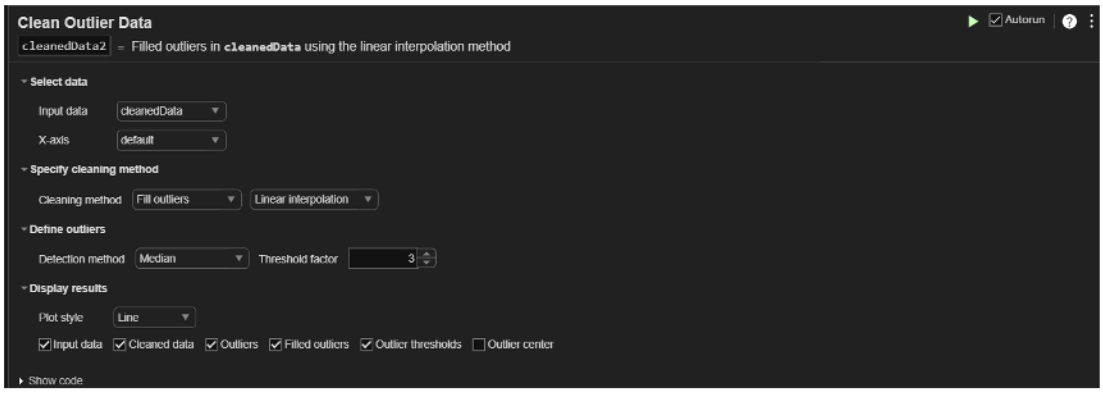
- ➢ Judul : Terdapat tulisan "Number of filled missing entries: 4" yang berarti ada 4 entri yang hilang dan telah diisi dengan nilai baru.
- ➢ Garis Biru : Garis biru menggambarkan data yang telah dibersihkan. Ini adalah data setelah proses *cleaning* dan pengisian nilai pada entri yang hilang.
- ➢ Titik Merah : Titik-titik merah menandakan posisi di mana entri yang hilang telah diisi dengan nilai baru. Ada 4 titik merah yang sesuai dengan 4 entri yang hilang.
- ➢ Fluktuasi Data : Pada grafik, ada fluktuasi yang signifikan, terutama pada posisi tertentu (misalnya, di sekitar indeks 10, 30, 70, dan 90). Kemungkinan, entri-entri hilang ini berada pada titik-titik dengan fluktuasi yang besar atau signifikan.
- ➢ Sumbu X : Sumbu X menunjukkan indeks data, yang bisa diartikan sebagai waktu atau urutan dari data itu sendiri.

- Sumbu Y : Sumbu Y menggambarkan nilai data, dengan beberapa entri memiliki nilai sangat tinggi atau rendah, seperti yang terlihat pada lonjakan dan penurunan di grafik.

Secara keseluruhan, grafik ini menunjukkan bagaimana menangani data yang hilang dengan metode pengisian dan menggambarkan hasil dari data yang sudah di-*cleaned* dibandingkan dengan data mentah.

C. FILL OUTLIERS

Berfungsi untuk menangani outlier (data yang menyimpang jauh dari data lainnya). Proses ini memungkinkan untuk mengisi outlier yang ada pada dataset menggunakan metode interpolasi linier.



Clean Outlier Data

Terdapat beberapa komponen pada Clean Outlier Data yaitu :

- **cleanedData2** = Hasil dari proses ini akan disimpan di variabel `cleanedData2`, yang merupakan data setelah outlier diisi. Data asli yang digunakan adalah `cleanedData`.
- **Select Data** ada dua pilihan yaitu :
  - Input data: kumpulan data utama yang akan diolah, tempat MATLAB mencari outlier berdasarkan deteksi yang sudah ditentukan (misalnya, nilai yang jauh dari median).
  - X-axis: Memberikan dimensi lain untuk menampilkan data dalam grafik, membantu menganalisis bagaimana nilai outlier tersebar di sepanjang waktu atau dalam konteks lainnya. Sumbu X juga memungkinkan interpretasi yang lebih mudah ketika melihat grafik hasil pembersihan outlier.

- **Specify Cleaning Method**

  Metode yang digunakan untuk membersihkan atau mengatasi outlier dalam dataset. Ini adalah langkah penting dalam pengolahan data, karena outlier (nilai ekstrem yang menyimpang jauh dari nilai rata-rata data) dapat memengaruhi hasil analisis.

  Cleaning Method: Bisa memilih cara untuk membersihkan data yang hilang. Ada tiga pilihan Fill outliner, remove outliner atau none. Berikut penjelasan dari beberapa metode yang ada :

  1. Constant Value (Nilai Konstan) : Dalam metode ini, outlier akan digantikan dengan nilai yang konstan, yang ditentukan sebelumnya. Misalnya, jika ingin mengisi semua outlier dengan angka "0" atau "100". Biasanya digunakan jika memiliki alasan spesifik untuk mengisi semua outlier dengan nilai tetap tertentu. Ini sering dilakukan jika nilai tersebut secara logis dapat menggantikan outlier.
  2. Convert to Missing (Ubah menjadi Hilang) : Outlier akan diubah menjadi nilai missing (hilang) akan dianggap sebagai `NaN` (Not a Number). Ini membuat outlier tidak digunakan dalam analisis lebih lanjut. Ini berguna jika ingin menunjukkan bahwa outlier ada tetapi tidak ingin mengisi atau menghapusnya, atau jika ingin melakukan pembersihan lebih lanjut pada data yang hilang.
  3. Center Value (Nilai Tengah) : Outlier akan digantikan dengan nilai tengah dataset, yang bisa berupa median atau rata-rata dari data yang ada. Digunakan jika ingin mengganti outlier dengan nilai yang lebih representatif dari keseluruhan dataset, tanpa bergantung pada data di sekitar outlier secara spesifik.
  4. Clip to Threshold Value (Batas Ambang : Jika outlier melampaui batas tertentu, outlier tersebut akan diubah ke nilai ambang batas (threshold). Jadi, jika ada outlier yang sangat besar atau sangat kecil, mereka akan dibatasi ke nilai maksimum atau minimum tertentu. Digunakan jika memiliki batasan khusus pada data dan ingin memastikan bahwa nilai tidak melebihi batas tertentu, misalnya dalam pengukuran fisik atau keuangan.
  5. Previous Value (Nilai Sebelumnya) : Outlier akan digantikan dengan nilai yang muncul sebelumnya dalam dataset. Berguna jika data memiliki pola temporal atau urutan, di mana menganggap bahwa nilai sebelumnya bisa menjadi estimasi yang baik untuk outlier.

6. Next Value (Nilai Selanjutnya) : Outlier akan digantikan dengan nilai yang muncul setelahnya dalam dataset. Sama seperti metode "Previous Value", tetapi dalam hal ini menggunakan nilai yang datang setelah outlier, yang juga berguna untuk data berurutan atau data waktu (time series).
7. Nearest Value (Nilai Terdekat) : Outlier akan diisi dengan nilai yang paling dekat dengannya, berdasarkan jarak dalam dataset. Berguna jika menginginkan estimasi yang mendekati outlier, tetapi tidak memerlukan interpolasi linier atau metode yang lebih kompleks.
8. Linear Interpolation (Interpolasi Linier : Nilai outlier diperkirakan menggunakan garis lurus yang menghubungkan dua titik data terdekat di sekitar outlier. Cocok untuk data yang memiliki tren atau kecenderungan linier. Ini adalah metode sederhana dan cepat yang sering digunakan untuk memperkirakan nilai yang hilang atau outlier.
9. Spline Interpolation (Interpolasi Spline) : Menggunakan fungsi spline untuk memperkirakan nilai outlier. Ini menciptakan kurva halus antara titik data untuk memberikan estimasi yang lebih akurat dibandingkan interpolasi linier. Lebih cocok untuk data yang memiliki tren non-linier, di mana interpolasi linier tidak cukup akurat.
10. Shape-Preserving Cubic Interpolation (PCHIP : Metode ini menggunakan interpolasi kubik tetapi mempertahankan bentuk data asli, sehingga nilai yang dihasilkan tidak melampaui nilai-nilai asli di sekitar outlier. Digunakan untuk data yang sangat bervariasi di mana interpolasi kubik biasa mungkin terlalu berlebihan dan menciptakan nilai yang melampaui batas data asli.
11. Modified Akima Cubic Interpolation : Ini adalah bentuk interpolasi kubik yang lebih kompleks yang menggunakan interpolasi segmen antara titik-titik data. Akima bekerja dengan mengurangi overshoot (peningkatan yang berlebihan) pada data yang bervariasi tajam. Berguna untuk data yang memiliki perubahan tajam atau tidak teratur, di mana interpolasi kubik biasa dapat menimbulkan nilai yang tidak realistis.

- **Define Outliers**

  Bagian Define Outliers bertujuan untuk mengidentifikasi dan menentukan batas nilai yang akan dianggap sebagai outlier dalam dataset. Ini adalah langkah penting sebelum memutuskan untuk membersihkan atau menangani outlier tersebut. Berikut penjelasan mengenai komponen-komponen yang ada di bagian ini:

- Detection Method (Metode Deteksi)

  Ini adalah metode yang digunakan untuk mengidentifikasi outlier dalam dataset. Beberapa metode yang sering digunakan di MATLAB antara lain:

  1. Median : Outlier dapat dideteksi dengan membandingkan nilai data terhadap median. Data yang jauh dari median bisa dianggap outlier. Median lebih tahan terhadap outlier daripada mean.
  2. Mean : Outlier dideteksi dengan membandingkan nilai data terhadap mean dan standar deviasi. Nilai yang berada jauh di luar batas (biasanya lebih dari 2 atau 3 standar deviasi dari mean) dianggap outlier.
  3. Quartiles : Deteksi outlier dilakukan menggunakan Interquartile Range (IQR). Data di luar batas atas dan bawah yang dihitung dari Q1 - 1.5*IQR atau Q3 + 1.5*IQR dianggap sebagai outlier.
  4. Grubbs : Digunakan untuk mendeteksi satu outlier pada satu waktu. Grubbs membandingkan nilai yang paling ekstrem (maksimum atau minimum) terhadap rata-rata dan standar deviasi. Nilai yang paling ekstrem dianggap outlier jika uji statistik Grubbs melebihi nilai kritis tertentu.
  5. Generalized Extreme Studentized Deviate (GESD) : Metode ini mendeteksi lebih dari satu outlier pada suatu waktu. GESD mengidentifikasi outlier dengan menggunakan statistik serupa Grubbs tetapi diadaptasi untuk lebih dari satu outlier.
  6. Moving Median : Digunakan dalam data time series. Outlier dapat dideteksi dengan membandingkan setiap titik data dengan median dari subset data yang bergerak (sliding window). Jika titik data berada jauh dari median, maka dapat dianggap outlier.
  7. Moving Mean : Seperti moving median, tetapi menggunakan rata-rata (mean) dari subset data yang bergerak. Outlier dideteksi dengan mengukur jarak antara titik data dan rata-rata bergerak, terutama jika jarak melebihi batas tertentu.
  8. Percentiles : Outlier dideteksi berdasarkan nilai persentil. Nilai yang jauh dari persentil tertentu (misalnya persentil ke-5 dan ke-95) dapat dianggap outlier.
  9. Range : Outlier bisa dideteksi dengan memeriksa nilai yang berada di luar rentang tertentu. Jika nilai data sangat jauh dari nilai minimum dan maksimum yang diharapkan, itu mungkin merupakan outlier.

10. Workspace Variable tidak secara langsung digunakan untuk mendeteksi outlier, tetapi merujuk pada variabel-variabel yang tersimpan selama sesi kerja di MATLAB. Variabel-variabel ini berisi data, hasil perhitungan, atau fungsi yang sedang digunakan.

- Threshold Factor (Faktor Ambang) : Faktor ambang ini digunakan untuk menentukan seberapa jauh nilai harus menyimpang dari pusat data (median atau mean) sebelum dianggap sebagai outlier.

➢ **Display Results**

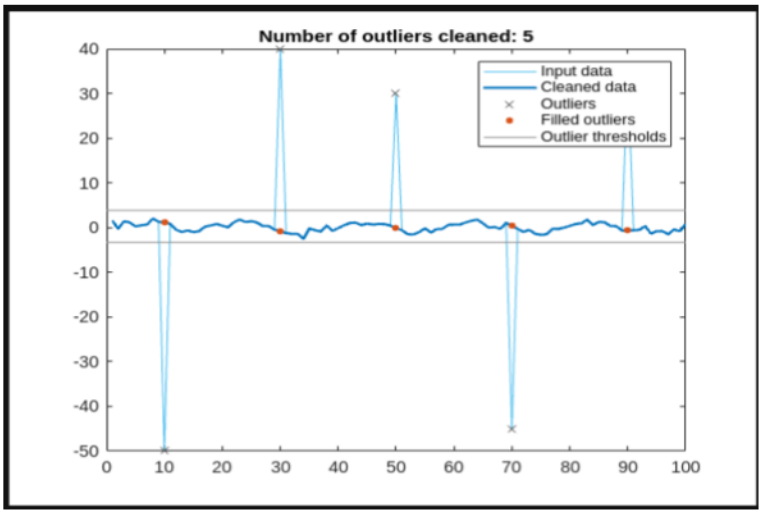
Visualisasi hasilnya yang ingin ditampilkan, ada dua plot style yaitu line dan histogram. Beberapa checkbox di bawahnya menentukan elemen-elemen apa saja yang akan divisualisasikan:

- Input data : Data asli (mentah) sebelum pembersihan.
- Cleaned data : Data setelah pembersihan.
- Outliers : Data yang terdeteksi sebagai outlier.
- Filled outliers : Outlier yang sudah diisi menggunakan interpolasi linier.
- Outlier thresholds : Menampilkan garis batas untuk mendeteksi outlier.
- Outlier center : Tidak dicentang, jadi pusat dari outlier tidak akan ditampilkan.

➢ **Show code**

Jika klik opsi ini, MATLAB akan menampilkan kode yang dihasilkan untuk proses ini, memungkinkan kamu melihat atau memodifikasi kode MATLAB yang terkait dengan tugas pengisian outlier.

Berikut hasil gambar setelah mengisi outliner yang sesuai dengan gambar :



Gambar grafik diatas merupakan hasil dari proses cleaning outliers pada data menggunakan MATLAB. Proses ini mengidentifikasi dan membersihkan outliers, yaitu data yang sangat menyimpang dari pola umum. Berikut penjelasan detailnya:

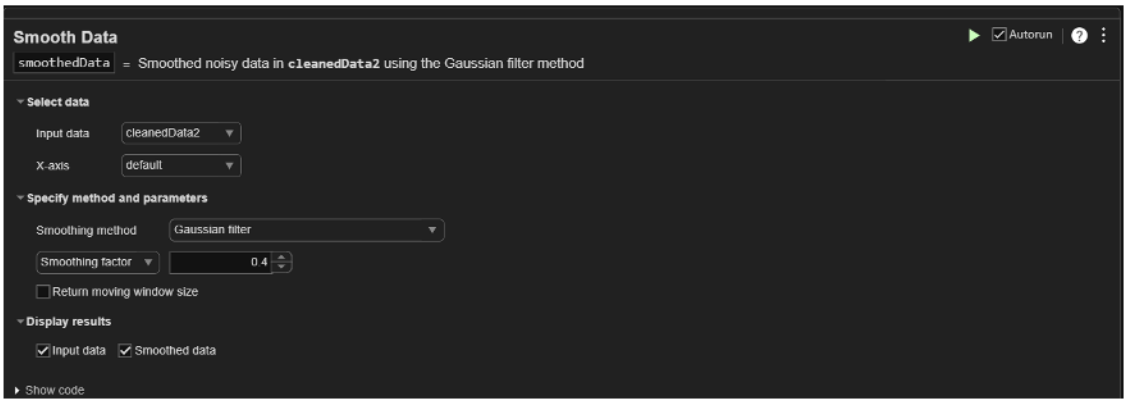
1. Judul : "Number of outliers cleaned: 5" menunjukkan bahwa ada 5 outliers yang telah diidentifikasi dan dibersihkan dari data.
2. Garis Biru (Input Data) : Ini menunjukkan data asli sebelum dilakukan pembersihan. Dari grafik, terlihat bahwa data ini memiliki beberapa nilai yang sangat tinggi atau rendah yang dianggap sebagai outliers.
3. Garis Biru Tipis Mendatar (Outlier Thresholds) : Garis ini menunjukkan batas ambang yang digunakan untuk mengidentifikasi outliers. Titik data yang berada jauh di luar ambang ini dianggap sebagai outliers.
4. Tanda X (Outliers) : Simbol X menunjukkan lokasi outliers dalam data. Outliers ini adalah nilai yang sangat menyimpang dari pola umum dan berada di luar batas ambang yang ditentukan.
5. Titik Merah (Filled Outliers) : Titik merah menunjukkan posisi outliers yang telah dibersihkan (diisi ulang) dengan nilai yang lebih sesuai. Ini berarti data di posisi tersebut tidak lagi berupa outliers, tetapi telah diisi dengan nilai baru.
6. Sumbu X : Sumbu X mewakili indeks data (urutan data), yang mungkin merepresentasikan waktu atau urutan pengamatan.
7. Sumbu Y : Sumbu Y mewakili nilai data. Terlihat bahwa outliers memiliki nilai yang sangat ekstrem, baik di bagian atas (nilai tinggi) maupun di bawah (nilai rendah), misalnya di sekitar indeks 20, 50, dan 90.

Secara keseluruhan, grafik ini menunjukkan proses di mana MATLAB mendeteksi dan mengoreksi outliers dalam dataset, menghasilkan data yang lebih bersih dan lebih mewakili pola umum.

D. SMOOTH DATA

Selanjutnya menghaluskan data yang sudah dibersihkan dari tugas sebelumnya dengan menggunakan tugas Smooth Data. Ketikkan kata kunci smooth dan klik tugas ketika muncul. Pilih data yang dibersihkan2, keluaran dari tugas sebelumnya, sebagai data masukan. Pilih

metode penghalusan, dan sesuaikan faktor penghalusan agar penghalusan lebih banyak atau lebih sedikit.



Berikut penjelasan dari komponen-komponen Smooth Data seperti gambar diatas :

- **Output Variable** : Variabel hasil smoothing diberi nama `smoothedData`. Ini adalah data hasil dari proses smoothing yang akan disimpan ke dalam variabel baru.
- Pada bagian **Select Data** terdapat dua pilihan yaitu :
  - Input Data : Pada bagian Input data, tempat memilih data mana yang akan di-smoothing. Disini memilih variabel `cleanedData2` sebagai data input. Ini adalah data yang akan di-smoothing.
  - X-axis : Disini diatur ke default, yang berarti MATLAB akan menggunakan sumbu X default dari data input.
- Pada bagian **Specify method and parameters** terdapat beberapa pilihan yaitu :
  - **Smoothing method,** ada beberapa metode smooth data yang bisa dipilih. Berikut penjelasan dari metode-metode tersebut :
    1. Moving Mean : Metode smoothing yang menggunakan rata-rata nilai dari sejumlah titik data sebelumnya dan sesudahnya dalam jendela bergerak. Ini adalah salah satu metode smoothing yang paling sederhana. Cocok untuk menghilangkan fluktuasi kecil atau noise dari data, tetapi bisa mengaburkan detail penting.
    2. Moving Median (Median Bergerak) : Menghitung nilai median dalam jendela bergerak, daripada rata-rata seperti pada moving mean. Metode ini lebih baik dalam menangani data yang memiliki outlier (nilai yang sangat berbeda), karena median kurang dipengaruhi oleh outlier dibandingkan rata-rata.

3. Gaussian Filter : Untuk menerapkan smoothing pada data. Metode ini memberikan bobot lebih besar pada titik data yang lebih dekat ke pusat jendela dan bobot lebih kecil pada titik yang lebih jauh. Cocok untuk data yang membutuhkan smoothing halus tanpa banyak mengaburkan detail, karena penurunan bobotnya gradual.
4. Local Linear Regression (Lowess) : Menggunakan regresi linear lokal untuk menerapkan smoothing pada data. Ini berarti bahwa data dalam jendela lokal dipasangkan dengan garis lurus yang sesuai dengan data di dalam jendela tersebut. Cocok untuk smoothing data yang memiliki pola yang tidak linier, karena metode ini fleksibel dan bisa menyesuaikan terhadap pola lokal.
5. Local Quadratic Regression (Loess) : Mirip dengan Lowess, tetapi menggunakan regresi kuadratik (polinomial orde dua) dalam jendela lokal. Metode ini lebih baik untuk data yang memiliki pola kurvilinier (melengkung) karena menggunakan fitting kuadratik, sehingga lebih presisi untuk pola yang lebih kompleks dibanding Lowess.
6. Robust Lowess : Versi dari Lowess yang lebih tahan terhadap outlier. Setelah fitting linear lokal dilakukan, proses iteratif menyesuaikan agar outlier tidak terlalu mempengaruhi hasil smoothing. Cocok jika data mengandung banyak outlier yang bisa mendistorsi hasil smoothing.
7. Robust Loess : Versi dari Loess yang lebih tahan terhadap outlier, menggunakan metode iteratif untuk mengurangi pengaruh outlier. Sangat berguna untuk data melengkung yang mengandung outlier, karena bisa mempertahankan pola melengkung sambil meminimalkan efek outlier.
8. Savitzky-Golay Polynomial Filter : Savitzky-Golay filter menggunakan fitting polinomial orde tertentu pada jendela data yang bergerak. Metode ini sangat cocok untuk menjaga bentuk asli dari sinyal (misalnya, mempertahankan puncak dan lembah) sambil menghaluskan noise. Cocok untuk data spektral atau sinyal di mana mempertahankan struktur asli sinyal sangat penting.

- Smoothing Factor dan Moving Window : Smoothing factor bisa mempengaruhi ukuran moving window, tergantung pada metode smoothing yang digunakan. Semakin besar smoothing factor, semakin lebar jendela bergerak, artinya lebih banyak titik data yang dipertimbangkan untuk setiap perhitungan nilai baru. Semakin kecil smoothing factor,

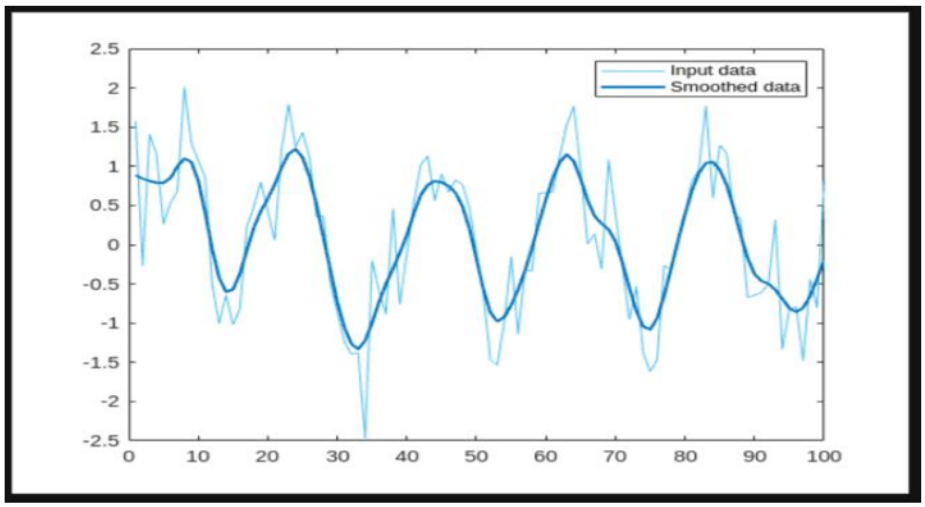
semakin kecil jendela, yang berarti smoothing akan lebih halus tetapi mempertahankan lebih banyak detail asli data.

- Return moving window size : Opsi ini tidak dicentang, artinya kamu tidak meminta MATLAB untuk menampilkan ukuran jendela bergerak yang digunakan dalam smoothing.

➢ **Display Results** : Pada bagian ini, Jika mencentang kotak Input data dan Smoothed data, yang berarti hasil smoothing akan dibandingkan dengan data input aslinya ketika ditampilkan.

➢ **Show Code** : Jika kamu mengklik opsi Show code, MATLAB akan menampilkan kode yang dihasilkan berdasarkan parameter yang telah kamu pilih.

Gambar ini menunjukkan bahwa sudah melakukan konfigurasi smoothing menggunakan filter Gaussian pada data `cleanedData2` dengan smoothing factor sebesar 0.4, dan berikut hasil smoothing yang didapat :



Gambar hasil smoothing data diatas menampilkan grafik dengan dua garis yang berbeda :

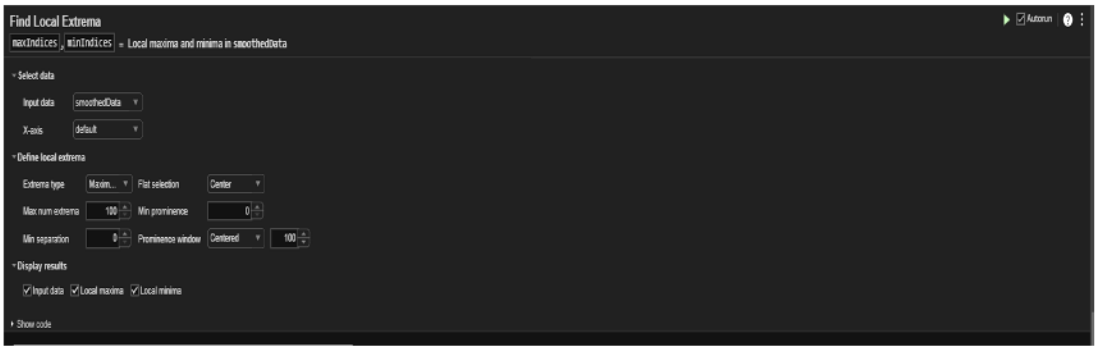
1. Input Data (Data Asli): Garis tipis berwarna biru muda yang berfluktuasi dengan tajam adalah input data. Data ini mungkin mengandung noise atau variasi yang cepat, sehingga terlihat sangat berfluktuasi dan tidak stabil.
2. Smoothed Data (Data yang Telah Di-smoothing): Garis tebal berwarna biru gelap menunjukkan smoothed data. Data ini telah melalui proses smoothing, sehingga fluktuasinya jauh lebih halus dan lebih konsisten dibandingkan input data.

Hasil smoothing ini mencoba mengurangi fluktuasi atau noise dari data asli untuk memberikan gambaran yang lebih stabil mengenai tren atau pola di data tersebut. Dalam grafik ini, smoothing telah berhasil menghaluskan fluktuasi data. Garis smoothed data mengikuti pola dasar dari input

data, tetapi dengan lebih sedikit variasi tajam. Dengan demikian akan memberikan hasil visual yang lebih mudah diinterpretasi untuk mendeteksi pola, misalnya dalam kasus tren naik turun, pergerakan musiman, atau pola berulang lainnya.

E. LOCATE EXTREMA

Terakhir, mulailah mengetik kata kunci ekstrim dan klik Find Ekstrema Lokal. Untuk menemukan titik-titik maksimum (puncak) dan minimum (lembah) lokal pada suatu data, dalam hal ini "smoothedData" (data yang sudah dihaluskan). Dapat menyesuaikan parameter ekstrem lokal untuk menemukan lebih banyak atau lebih sedikit maksimum dan minimum.



Berikut penjelasan dari komponen-komponen Locate Extrema pada gambar diatas :

- Pada bagian **Select Data** terdapat dua pilihan yairu :
  - Input Data : Menunjukkan bahwa data yang akan dianalisis, disini pada variabel "smoothedData".
  - X-axis : Sumbu X default yang digunakan untuk memplot data.

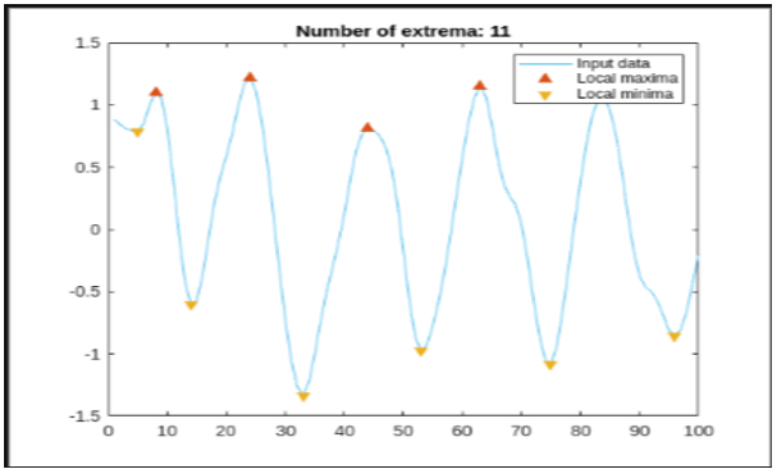
2. **Define Local Extrema** terdapat beberapa pilihan yaitu :
   - Extreme Type : Memilih apakah ingin mencari Maxima (titik maksimum), Minima (titik minimum), atau keduanya.
   - Flat Selection : Terdapat empat pilihan: Center, First, Last, dan All. Opsi Center akan memilih titik tengah dari dataran (flat) untuk diidentifikasi sebagai puncak atau lembah lokal. First memilih titik awal dari dataran datar sebagai ekstrem. Last akan memilih titik terakhir dari dataran datar sebagai ekstrem. Sementara itu, All akan mengidentifikasi semua titik pada dataran datar sebagai maksimum atau minimum lokal, tergantung pada konteksnya.

- Max num extrema : Batas jumlah titik ekstrem yang akan dicari, disini diatur ke 100 pada gambar.
- Min Prominence : Nilai minimum yang menunjukkan seberapa mencolok suatu puncak atau lembah. Diatur ke 0, artinya semua puncak atau lembah akan dipertimbangkan tanpa pembatasan berdasarkan prominensinya.
- Min Separation : Jarak minimum antara dua titik ekstrem yang diatur ke 0, yang berarti tidak ada pembatasan jarak.
- Prominence Window : Opsi untuk mengatur bagaimana prominensi (perbedaan ketinggian puncak/lembah terhadap tetangganya) dihitung, disini pada gambar diatur ke "Centered" atau simetris dengan lebar jendela 100. Jika diatur ke “Asymmetric”menghitung prominensi (kejelasan/ketinggian relatif) puncak atau lembah menggunakan jendela yang tidak simetris di sekitar titik tersebut.

➢ **Display Results** : ada tiga pilihan yaitu Input data, Local maxima, Local minima. Opsi ini untuk menampilkan data input, maksimum lokal, dan minimum lokal pada hasil.

➢ **Show Code** : Bagian ini biasanya akan menunjukkan kode MATLAB yang sesuai dengan pilihan di atas jika diaktifkan, memungkinkan pengguna untuk menghasilkan kode untuk keperluan pemrograman otomatis.

Hasil yang didapat jika Locate Extrema diatur sesuai pilihan yang ada pada gambar :



Gambar diatas menunjukkan grafik hasil dari pencarian ekstrema lokal pada data input menggunakan fitur Locate Extrema. Berikut penjelasan dari hasil tersebut :

1. Grafik Sinusoidal : Garis biru pada grafik menunjukkan data input yang dianalisis. Ini tampaknya adalah fungsi sinusoidal atau data periodik yang bergelombang dengan puncak dan lembah.
2. Local Maxima (Titik Maksimum Lokal) : Ditandai dengan segitiga merah menghadap ke atas (▲), titik-titik ini menunjukkan lokasi maksimum lokal, yaitu titik di mana nilai data mencapai puncak tertinggi dibandingkan tetangganya.
3. Local Minima (Titik Minimum Lokal) : Ditandai dengan segitiga kuning menghadap ke bawah (▼), titik-titik ini menunjukkan lokasi minimum lokal, yaitu titik di mana nilai data mencapai lembah terendah dibandingkan tetangganya.
4. Jumlah Ekstrema : Pada bagian atas grafik tertulis "Number of extrema: 11", yang berarti terdapat 11 titik ekstrem, termasuk maksimum dan minimum lokal, yang terdeteksi pada data tersebut.

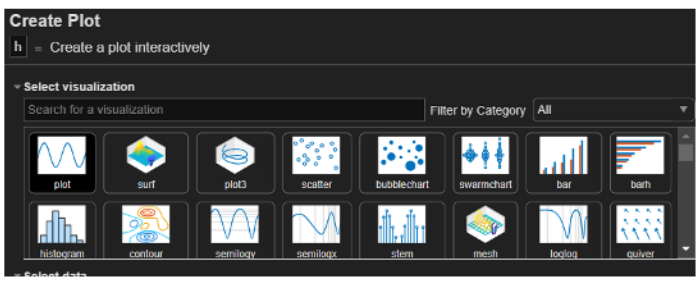
Secara keseluruhan, hasil ini menunjukkan identifikasi titik-titik puncak dan lembah dari data input yang dianalisis.

F. UJI COBA

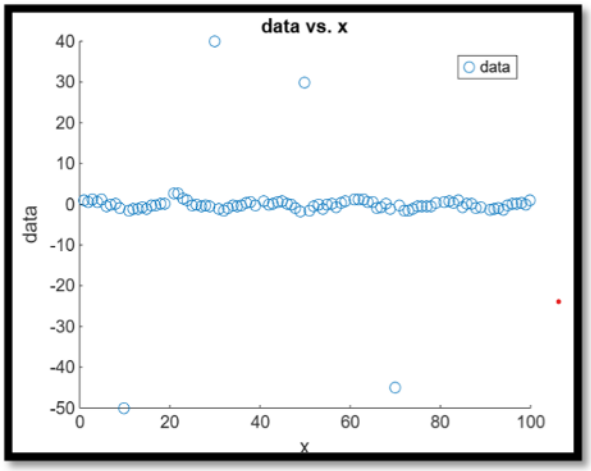
Selanjutnya akan melakukan uji coba untuk melihat tampilan gambar visualisasi yang berbeda dengan plot data vektor yang sama :

```
x = 1:100;
data = cos(2*pi*0.05*x + 2*pi*rand) + 0.5*randn(1,100);
data(20:20:80) = NaN;
data(10:20:90) = [-50 40 30 -45 35];
```

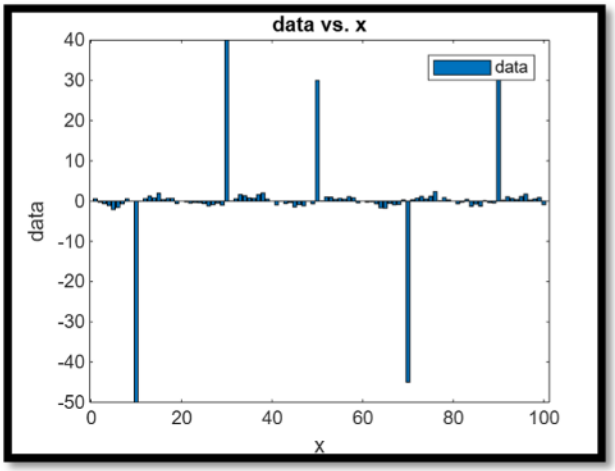
Percobaan pertama akan mengubah visualisasi pada **Create Plot** untuk melihat tampilan visualisasi yang berbeda :



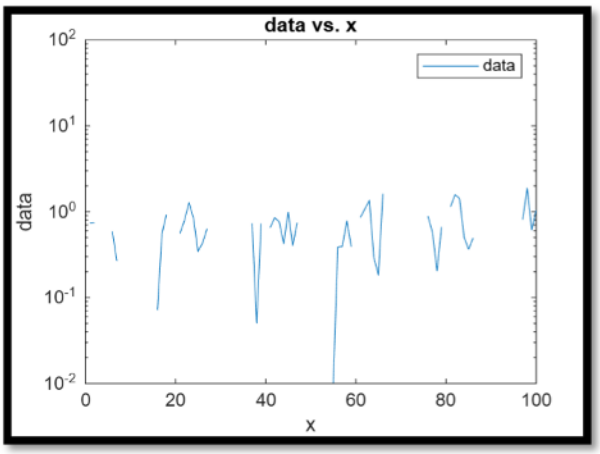
Berikut hasil gambar yang didapat :



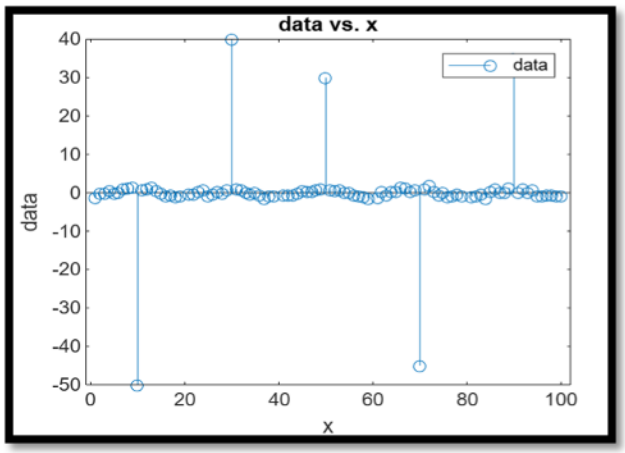
Scatter



Bar



Semilogy



Stem

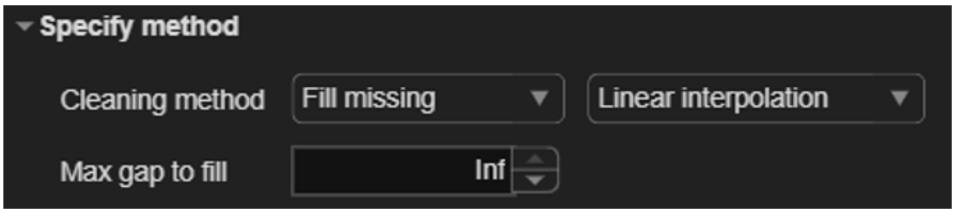
Bisa dilihat dari hasil gambar diatas, terdapat empat visualisasi dengan 4 tampilan gaya yang berbeda meskipun data yang di plot sama dengan dua variabel yaitu variabel data dan x, berikut penjelasannya :

1. **Scatter**: Menampilkan titik-titik data secara individual, cocok untuk melihat distribusi atau pola outlier. Ideal untuk mengevaluasi korelasi antara dua variabel atau mendeteksi anomali. Dalam contoh ini, titik-titik tersebar pada sumbu x dan y.
2. **Bar** : Data divisualisasikan sebagai batang vertikal, cocok untuk perbandingan kategori atau nilai diskrit. Digunakan untuk menunjukkan nilai dalam bentuk batang, sehingga mempermudah membandingkan jumlah atau frekuensi antar kategori atau interval.

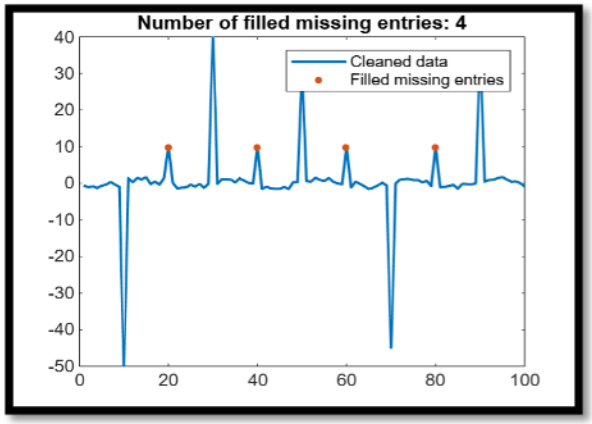
3. **Semilogy** : Sumbu y menggunakan skala logaritmik, efektif untuk melihat data yang memiliki variasi nilai besar atau yang tumbuh secara eksponensial. Skala logaritmik pada sumbu y membantu menampilkan data yang berkisar sangat besar dengan lebih jelas, sehingga perbedaan kecil maupun besar tetap terlihat.
4. **Stem** : Memvisualisasikan data dengan garis vertikal yang terhubung ke titik, berguna untuk menyoroti nilai diskrit dalam deret. Representasi yang mirip dengan bar namun lebih sederhana, sering digunakan untuk sinyal diskrit atau data time series.

Keempat visualisasi ini memberikan perspektif berbeda pada data yang sama tergantung tujuan analisisnya.

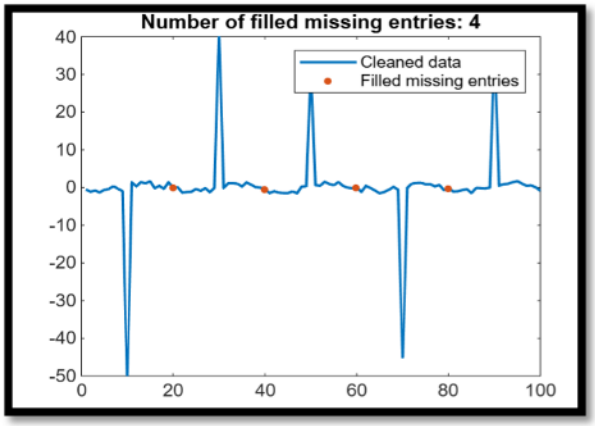
Selanjutnya akan dilakukan uji coba pada **Fill Missing Data** dengan specify method.



Berikut dua hasil dari uji coba tersebut :



Constant Value : 10



Moving Mean (Moving window : Centered, 10)

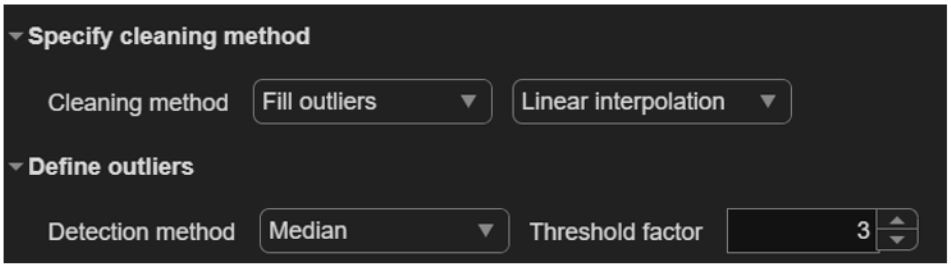
Diatas merupakan dua contoh **Fill Missing Data** dengan cleaning method yang berbeda. Berikut penjelasannya :

1. **Constant Value: 10** (Grafik kiri) : Metode ini mengisi nilai yang hilang dengan nilai konstan, dalam hal ini, 10. Pada grafik terlihat bahwa titik-titik yang hilang diisi dengan nilai 10 (ditandai dengan titik oranye) sehingga titik yang diisi terlihat lebih seragam. Keunggulannya

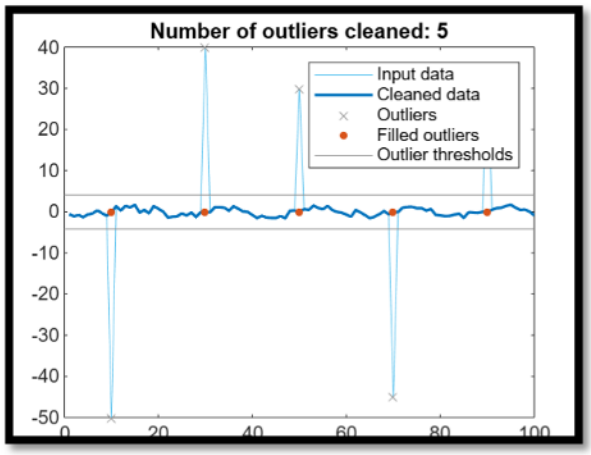
Sederhana dan cepat namun kelemahannya tidak memperhitungkan pola atau tren di sekitar titik hilang, sehingga dapat mengabaikan fluktuasi yang ada dalam data asli.

2. **Moving Mean (Moving window: Centered, 10)** : Metode ini mengisi nilai yang hilang menggunakan rata-rata dari jendela bergerak (moving window) yang berpusat pada titik yang hilang. Jendela berukuran 10 berarti metode ini menghitung rata-rata dari 10 nilai di sekitar titik yang hilang (5 sebelum dan 5 setelahnya). Keunggulannya Lebih adaptif terhadap pola atau tren dalam data di sekitar titik yang hilang. Kelemahannya Bisa menjadi tidak efektif jika datanya sangat bervariasi atau jika banyak outlier dalam jendela yang digunakan.

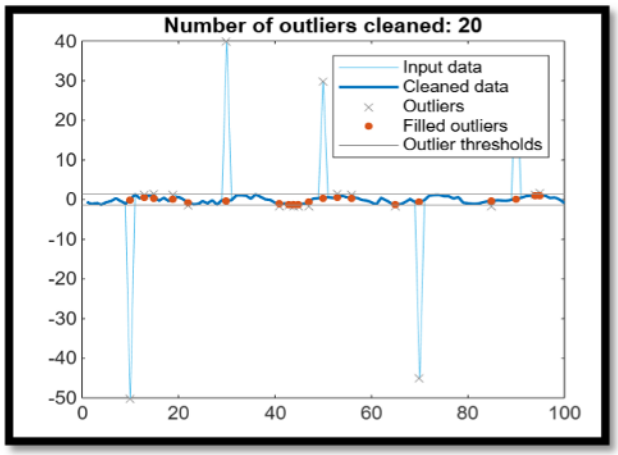
Percobaan berikutnya akan mengubah **Fill Outliers** pada bagian Specify Cleaning Method dan Define Outliner.



Berikut dua contoh gambar hail perubahan bagian Specify Cleaning Method dan Define Outliner :



Center Value (Quartiles, threshold : 2)



Modified Akima Cubic Interpolation (Percetiles, Lower threshold : 10, Upper threshold : 90)

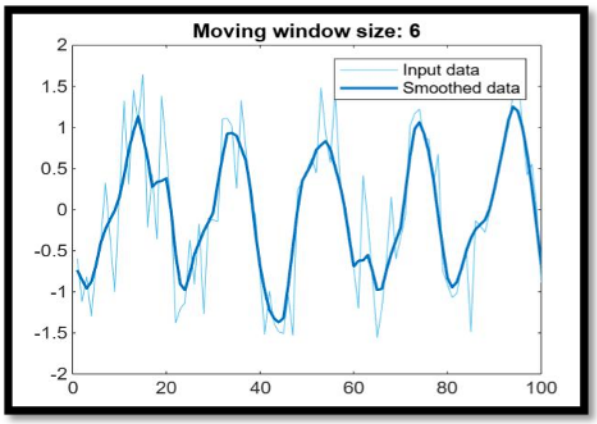
Pada kedua gambar diatas, proses Fill Outlier digunakan untuk membersihkan data outlier dengan dua metode berbeda :

1. **Center Value (Quartiles, threshold: 2)** : Metode ini mendeteksi outlier berdasarkan kuartil (pembagian data menjadi empat bagian). Outlier ditentukan menggunakan ambang batas "threshold: 2", dan data yang berada jauh di luar batas dihitung sebagai outlier. Outlier dihilangkan dan diisi dengan nilai yang lebih mendekati tren data lainnya (titik merah menunjukkan outlier yang telah diisi).
2. Modified Akima Cubic Interpolation (Percentiles, Lower threshold: 10, Upper threshold : 90) : Menggunakan interpolasi kubik Akima yang dimodifikasi untuk mendeteksi dan mengisi outlier. Ambang batas bawah adalah 10% dan atas 90%, yang berarti outlier berada di luar rentang ini. Data yang diisi dengan interpolasi kubik memberikan hasil yang lebih halus dan akurat, khususnya untuk data dengan lebih banyak outlier.

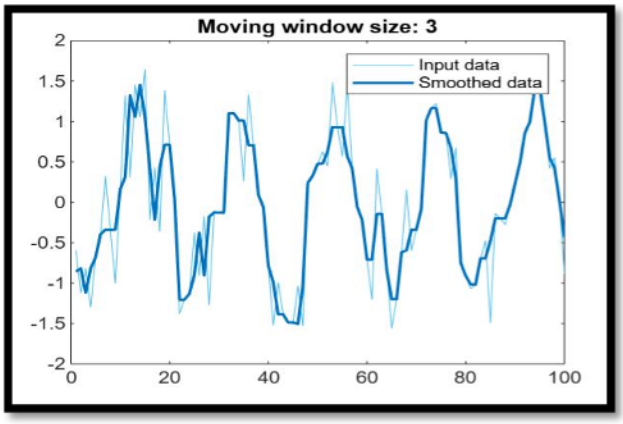
Jadi, dua pendekatan ini membersihkan outlier dengan teknik berbeda, satu berdasarkan kuartil, dan yang lain interpolasi kubik dengan batasan persentil.

Selanjutnya akan dilakukan uji coba perubahan komponen pada **Smoothing Data**.

Berikut dua hasil uji coba yang dilakukan pada Smoothind Data dengan mengubah komponen Specify method and parameters :



Robust Lowess(Moving Window : Center 6)



Moving Median (Smoothing Factor : 0,2)

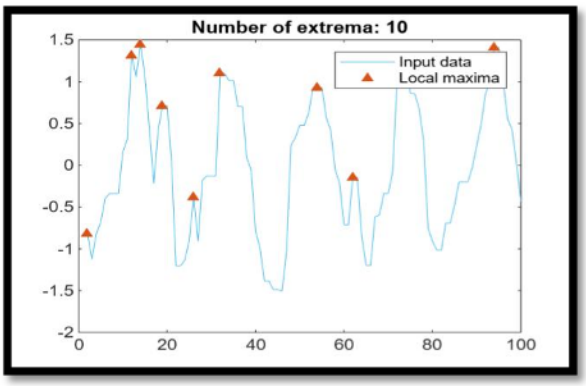
Pada gambar diatas, terdapat dua hasil smoothing data(penghalusan data) dari input yang sama menggunakan dua metode berbeda :

1. **Robust Lowess (Moving Window: Center 6)** : Grafik menggunakan metode Lowess (Locally Weighted Scatterplot Smoothing) dengan jendela bergerak (moving window) berukuran 6. Metode ini melakukan fitting lokal pada data untuk menghasilkan garis yang lebih halus. Penggunaan Robust berarti metode ini lebih tahan terhadap outlier (data yang menyimpang jauh). Semakin besar ukuran window, semakin halus hasil smoothing.
2. **Moving Median (Smoothing Factor: 0,2)** : Grafik di sebelah kanan menggunakan metode moving median dengan smoothing factor 0,2. Ini adalah metode berbasis median di mana median dihitung dalam jendela bergerak untuk menghaluskan data. Smoothing factor menentukan seberapa besar dampak smoothing terhadap data asli.

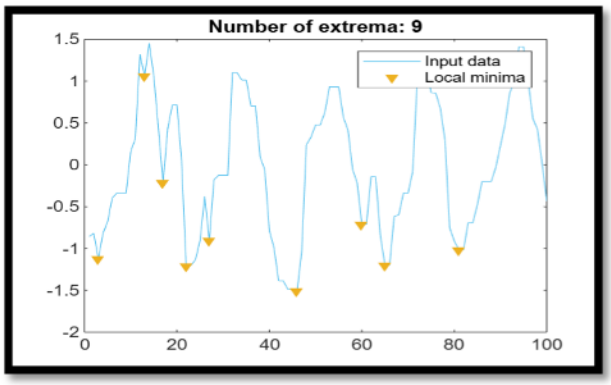
Kedua grafik menunjukkan data input dan hasil penghalusannya, dengan yang kiri lebih halus (lebih kuat terhadap outlier) dibanding yang kanan.

Terakhir kita akan melakukan uji coba perubahan komponen pada **Locate Extrema**.

Perubahan komponen hanya dilakukan pada Extreme type menjadi Maxima dan Minima. Sehingga menghasilkan dua gambar dibawah ini :



Maxima



Minima

Dalam gambar tersebut, kita melihat dua plot yang menunjukkan hasil dari pencarian ekstrem pada data input yang sama. Perbedaan antara kedua plot tersebut adalah jenis ekstrem yang ditemukan:

1. **Maxima** : Menunjukkan local maxima, yaitu titik-titik pada grafik di mana nilai data berada pada puncak lokal. Dengan kata lain, ini adalah titik-titik di mana nilai data lebih besar dari tetangga terdekatnya di kiri dan kanan.
2. **Minima** (plot kanan): Menunjukkan local minima, yaitu titik-titik pada grafik di mana nilai data berada pada dasar lokal. Ini adalah titik-titik di mana nilai data lebih kecil dari tetangga terdekatnya di kiri dan kanan.