Point Cloud Distance Analysis (M3C2 Output)

Fundamental Metrics

Dataset Overview

- **Total Count**: Total number of distance values (including NaN). Provides context for dataset size.

  ```
  total_count = len(distances)
  ```

- **NaN Count**: Number of invalid/failed computations (e.g., no neighbors found within search radius). Lower values indicate better coverage.

  ```
  nan_count = int(np.isnan(distances).sum())
  ```

  - `np.isnan()` returns a Boolean array (`True` where NaN exists)
  - `.sum()` counts the total number of NaN values

- **% NaN**: Proportion of failed computations. High percentages indicate poor coverage or inadequate parameter settings.

  ```
  perc_nan = (nan_count / total_count) * 100 if total_count > 0 else np.nan
  ```

- **% Valid**: Proportion of successful computations (complement of % NaN). Higher values indicate robust coverage.

  ```
  perc_valid = ((total_count - nan_count) / total_count) * 100 if total_count
  > 0 else np.nan
  ```

- **Valid Count**: Number of non-NaN distance values after optional range clipping.

  ```
  valid = distances[~np.isnan(distances)]
  clipped = valid[(valid >= data_min) & (valid <= data_max)]
  valid_count = int(clipped.size)
  ```

  - `range_override`: Optional tuple `(min, max)` to explicitly set the analysis range
  - If not specified, `data_min`/`data_max` are computed from the data

- **Valid Sum**: Sum of all valid distance values.

- Near zero → deviations cancel out → no systematic bias between clouds
- Positive → comparison surface is systematically above/outside the reference
- Negative → comparison surface is systematically below/inside the reference

```
valid_sum = float(np.sum(clipped))
```

- **Valid Squared Sum**: Sum of squared valid distance values.
  - Each distance d_i is squared (d_i^2), then summed: $\sum_{i=1}^{n} d_i^2$
  - Always non-negative
  - Heavily influenced by outliers due to squaring

```
valid_squared_sum = float(np.sum(clipped ** 2))
```

M3C2 Parameters

- **Normal Scale**: Radius (in point cloud units) used for local surface normal estimation.
  - Too small → noise dominates, unstable normals
  - Too large → over-smoothing, loss of local detail
  - Typically set to capture local surface geometry while filtering noise

```
normal_scale  # User-defined parameter
```

- **Search Scale**: Radius of the projection cylinder along the normal direction.
  - Rule of thumb: ~2× Normal Scale
  - Too small → few/no points found → many NaN values
  - Too large → excessive smoothing, loss of detail

```
search_scale  # User-defined parameter
```

Location & Dispersion Metrics

Central Tendency

- **Min / Max**: Extreme distance values in the dataset. Useful for identifying outliers and data range.

```
min_val = float(np.nanmin(distances))
max_val = float(np.nanmax(distances))
```

- **Mean (Bias)**: Arithmetic mean of distances. Ideally near zero for unbiased comparisons.

  $$\bar{d} = \frac{1}{n} \sum_{i=1}^{n} d_i$$

  ```
  avg = float(np.mean(clipped))
  ```

- **Median**: Robust measure of central tendency, less sensitive to outliers than mean.

  ```
  med = float(np.median(clipped))
  ```

Spread Measures

- **Empirical Standard Deviation**: Measure of dispersion around the mean. Sensitive to outliers.

  $$\sigma = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (d_i - \bar{d})^2}$$

  ```
  std_empirical = float(np.std(clipped, ddof=1))  # Note: ddof=1 for sample std
  ```

- **RMS (Root Mean Square)**: Combined measure of bias and spread.

  $$\text{RMS} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} d_i^2}$$

  - Includes both systematic offset (bias) and random variation (spread)
  - Always ≥ |Mean| (equality when all values are identical)

  ```
  rms = float(np.sqrt(np.mean(clipped ** 2)))
  ```

- **MAE (Mean Absolute Error)**: Average magnitude of deviations, robust to outliers.

  $$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |d_i|$$

  - More robust than RMS due to linear (not quadratic) penalty
  - MAE = 0 → perfect agreement
  - MAE = 0.01 m → average deviation of 1 cm between clouds

  ```
  mae = float(np.mean(np.abs(clipped)))
  ```

- **NMAD (Normalized Median Absolute Deviation)**: Robust standard deviation estimator.

  $$\text{NMAD} = 1.4826 \times \text{median}(|d_i - \text{median}(d)|)$$

- Factor 1.4826 makes NMAD equivalent to σ for normal distributions
- Highly robust to outliers (50% breakdown point)

```
mad = float(np.median(np.abs(clipped - med)))
nmad = float(1.4826 * mad)
```

Inlier/Outlier Analysis

Classification Criteria

- **Outlier Definition**: Points with |distance| > 3×RMS
- **Inlier Definition**: Points with |distance| ≤ 3×RMS

Subset Statistics

- **MAE Inlier**: Mean absolute error computed only for inliers.

```
mae_in = float(np.mean(np.abs(inliers))) if inliers.size > 0 else np.nan
```

- **NMAD Inlier**: Robust spread measure for inliers only.

```
median_inliers = np.median(inliers)
nmad_in = float(1.4826 * np.median(np.abs(inliers - median_inliers)))
         if inliers.size > 0 else np.nan
```

- **Outlier/Inlier Counts**:
  - Total outliers and inliers (sum equals valid_count)
  - Positive/negative outliers: Points above/below zero
  - Positive/negative inliers: Distribution of inliers around zero
- **Mean/Std Statistics**:
  - Computed separately for inlier and outlier subsets
  - Useful for understanding systematic patterns in outliers

Quantile Statistics

- **Q05/Q95**: 5th and 95th percentiles
  - Range containing central 90% of data
  - More robust than min/max for identifying typical range
- **Q25/Q75**: First and third quartiles

- Interquartile Range (IQR) = Q75 - Q25
- Robust measure of spread, unaffected by outliers

Distribution Fitting

Gaussian (Normal) Distribution Fit

Fits a normal distribution $\mathcal{N}(\mu, \sigma^2)$ to the data using maximum likelihood estimation.

- **Gaussian Mean (μ)**: Location parameter of the fitted distribution

```
mu, std = norm.fit(clipped)
```

- **Gaussian Std (σ)**: Scale parameter of the fitted distribution

```
from scipy.stats import norm
mu, std = norm.fit(clipped)
```

Gaussian Chi-Square Goodness-of-Fit

Measures how well the data follows a normal distribution using Pearson's χ^2 test.

- **Low χ^2** → Data closely follows Gaussian distribution
- **High χ^2** → Significant deviations (skewness, heavy tails, multimodality)

Calculation steps:

1. **Compute expected frequencies under Gaussian model:**

```
# CDF at bin edges
cdf_left = norm.cdf(bin_edges[:-1], mu, std)
cdf_right = norm.cdf(bin_edges[1:], mu, std)

# Expected counts per bin
expected_gauss = N * (cdf_right - cdf_left)
```

2. **Filter bins with very low expected counts** (to avoid numerical instability):

```
min_expected = 1e-12  # or user-defined threshold
mask = expected_gauss > min_expected
```

3. **Calculate Pearson χ^2 statistic:**

$$\chi^2 = \sum_{i} \frac{(O_i - E_i)^2}{E_i}$$

where O_i = observed frequency, E_i = expected frequency

```
chi2_gauss = float(np.sum((hist[mask] - expected_gauss[mask])**2
                          / expected_gauss[mask]))
```

Weibull Distribution Fit

The Weibull distribution is particularly suitable for modeling skewed error distributions common in point cloud comparisons.

- **Probability Density Function:**

  $$f(x; k, \lambda, \theta) = \frac{k}{\lambda}\left(\frac{x-\theta}{\lambda}\right)^{k-1} e^{-\left(\frac{x-\theta}{\lambda}\right)^k}$$

  where:

  - k = shape parameter
  - λ = scale parameter
  - θ = location (shift) parameter

Weibull Parameters

- **Shape Parameter (k or a)**:

  - $k < 1$: Heavy right tail, exponential-like decay
  - $k = 2$: Rayleigh distribution
  - $k > 3.5$: Approaching normal distribution
  - Controls the distribution's asymmetry and tail behavior

  Weibull Shape Parameter Effect

- **Scale Parameter (λ or b)**:

  - Controls the width/spread of the distribution
  - Larger values → broader distribution
  - Roughly corresponds to a "stretching" of the distance distribution

  Weibull Scale Parameter Effect

- **Location Parameter (θ or loc)**:

  - Shifts the distribution along the x-axis
  - Often close to the minimum value for distance data
  - In CloudCompare, typically near the median or minimum depending on dataset

  Weibull Location Parameter Effect

```python
from scipy.stats import weibull_min
a, loc, b = weibull_min.fit(clipped)  # a=shape, loc=location, b=scale
```

Weibull-Derived Metrics

- **Mode**: Position of maximum probability density

  $$\text{Mode} = \begin{cases} \theta + \lambda\left(\frac{k-1}{k}\right)^{1/k} & \text{if } k > 1 \\ \theta & \text{if } k \leq 1 \end{cases}$$

- **Skewness**: Measure of asymmetry
  - Positive: Right-skewed (long right tail)
  - Negative: Left-skewed (long left tail)

- **Weibull χ^2**: Goodness-of-fit test, calculated analogously to Gaussian χ^2

Distribution Characteristics

- **Skewness**: Third standardized moment, measures asymmetry

  $$\text{Skewness} = \frac{\mathbb{E}[(X-\mu)^3]}{\sigma^3}$$

  - = 0: Symmetric distribution
  - > 0: Right-skewed (tail extends right)
  - < 0: Left-skewed (tail extends left)

- **Excess Kurtosis**: Fourth standardized moment minus 3, measures tail heaviness

  $$\text{Excess Kurtosis} = \frac{\mathbb{E}[(X-\mu)^4]}{\sigma^4} - 3$$

  - = 0: Normal distribution tails
  - > 0: Heavy tails (leptokurtic)
  - < 0: Light tails (platykurtic)

Tolerance & Coverage Metrics

- **% |Distance| > Threshold**: Fraction of points exceeding a specified tolerance (e.g., 1 cm)

- **% Within ±2σ**: Fraction within two standard deviations
  - ~95% for normally distributed data
  - Deviations indicate non-normality

- **Max |Distance|**: Maximum absolute deviation
  - Highly sensitive to outliers

  - Useful for worst-case analysis

- **Within Tolerance**: Fraction of values within user-defined tolerance bounds

---

Agreement & Comparison Metrics

Bland-Altman Analysis

Used to assess agreement between two measurement methods or point clouds. Answers the question: **"Do both methods provide comparable results?"**

Structure of the Bland-Altman Plot:

Bland-Altman Plot Example

- **x-axis (Mean of measurements)**: Mean of both methods per data point - shows the magnitude of measured values
- **y-axis (Difference)**: Difference between methods (typically Method A − Method B) - shows deviation magnitude and direction
- **Red line (Bias)**: Mean difference across all points - indicates systematic offset
- **Green lines (Limits of Agreement)**: Bias ± 1.96 × SD - range containing ~95% of differences

Key Components:

- **Bias (Mean Difference)**: Systematic offset between methods

  $\text{Bias} = \bar{d} = \frac{1}{n}\sum_{i=1}^{n} d_i$

- **Limits of Agreement (LoA)**: Expected range for 95% of differences

  $\text{LoA} = \text{Bias} \pm 1.96 \times \sigma_d$

  where σ_d is the standard deviation of differences

Interpretation:

- **Bias ≈ 0**: Methods agree on average
- **Bias ≠ 0**: One method consistently yields higher/lower results
- **Narrow LoA**: High agreement (low variance)
- **Wide LoA**: High uncertainty, poor reproducibility
- **Random scatter**: Differences independent of measurement magnitude (good)
- **Patterns/trends**: Systematic errors or heteroscedasticity (problematic)

References:

- Wikipedia: Bland–Altman plot
- PMC Article
- Datatab Tutorial

Passing-Bablok Regression

Non-parametric method for comparing measurement methods, robust to outliers.

Method:

1. Calculate slopes for all point pairs
2. Take median slope (β_1) and median intercept (β_0)
3. Compute confidence intervals

Interpretation:

- $1 \in$ CI(slope) AND $0 \in$ CI(intercept): Methods are comparable
- $1 \notin$ CI(slope): Proportional difference exists
- $0 \notin$ CI(intercept): Systematic difference exists

Single-Cloud Statistics

Input Parameters

- **Radius [m]**: Search radius for local neighborhood analysis
  - Larger: Smoother metrics, less noise-sensitive
  - Smaller: More detailed, captures fine-scale variation
- **k-NN**: Number of nearest neighbors for distance calculations
  - Larger k: More stable but less localized metrics
- **Sampled Points**: Number of randomly sampled points for computationally intensive metrics
  - Balances accuracy with computation time
- **Area Source**: Method for XY area estimation
  - `convex_hull`: Convex hull of 2D footprint (more realistic)
  - `bbox`: Axis-aligned bounding box (overestimate)

Global Height Statistics (Z-dimension)

- **Z Min/Max [m]**: Elevation extrema

  $$z_{\min} = \min(z), \quad z_{\max} = \max(z)$$

- **Z Mean/Median [m]**: Central tendency of elevation

  $$\bar{z} = \frac{1}{N}\sum_{i=1}^{N} z_i$$

  - Large mean-median difference indicates skewed distribution

- **Z Standard Deviation [m]**: Elevation variability

  $$\sigma_z = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N} (z_i - \bar{z})^2}$$

  - High: Strong relief variation

    - Low: Flat/homogeneous surface

- **Z Quantiles [m]**: Robust elevation range descriptors

    - Q05/Q95: Range excluding extreme 10%
    - Q25/Q75: Interquartile range (IQR)

Density Metrics

- **Global Density [pts/m²]**: Overall point density

  $$\rho_{\text{global}} = \frac{N}{A_{XY}}$$

  where N = point count, A_{XY} = footprint area

- **Local Density [pts/m³]**: Neighborhood-based density

  $$\rho_{\text{local}}(p) = \frac{|\mathcal{N}_r(p)|}{V_{\text{sphere}}}$$

  where $|\mathcal{N}_r(p)|$ = neighbor count, $V_{\text{sphere}} = \frac{4}{3}\pi r^3$

---

k-Nearest Neighbor Statistics

- **Mean Distance to 1st-kth NN [m]**: Average distance to k nearest neighbors

  $$\bar{d}_{1:k} = \frac{1}{M}\sum_{p \in S} \left(\frac{1}{k}\sum_{j=1}^{k} d_j(p)\right)$$

- **Mean Distance to kth NN [m]**: Scale indicator

  $$\bar{d}_k = \frac{1}{M}\sum_{p \in S} d_k(p)$$

    - Smaller values indicate denser sampling

Surface Roughness

- **Roughness [m]**: Standard deviation of points from local best-fit plane
    - Low: Smooth/planar surface
    - High: Rough, uneven, or noisy surface
    - Computed via PCA of local neighborhoods

PCA Shape Descriptors

Based on eigenvalue analysis of local point neighborhoods with eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq 0$:

- **Linearity**: Degree of linear structure

  $$L = \frac{\lambda_1 - \lambda_2}{\lambda_1}$$

    - High: Edge-like features

- **Planarity**: Degree of planar structure

  $$P = \frac{\lambda_2 - \lambda_3}{\lambda_1}$$

  - High: Surface-like features

- **Sphericity**: Degree of isotropic distribution

  $$S = \frac{\lambda_3}{\lambda_1}$$

  - High: Volumetric/scattered points

- **Anisotropy**: Overall directional bias

  $$A = \frac{\lambda_1 - \lambda_3}{\lambda_1}$$

  - High: Strong directional structure

- **Omnivariance [m^2]**: Geometric mean of eigenvalues

  $$O = (\lambda_1 \lambda_2 \lambda_3)^{1/3}$$

  - Scale-dependent measure of overall variance

- **Eigenentropy**: Disorder measure

  $$H = -\sum_{i=1}^{3} p_i \log(p_i), \quad p_i = \frac{\lambda_i}{\sum_j \lambda_j}$$

  - High: Disordered/isotropic
  - Low: Ordered/structured

- **Curvature**: Surface curvature indicator

  $$\kappa = \frac{\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3}$$

  - High: Strong curvature
  - Low: Flat surface

Orientation Metrics

- **Verticality [degrees]**: Angle between local normal and vertical (Z-axis)

  $$\theta = \arccos(|n_z|) \times \frac{180°}{\pi}$$

  - 0°: Horizontal surface (normal vertical)
  - 90°: Vertical surface (normal horizontal)

- **Normal Standard Deviation [degrees]**: Consistency of normal orientations

  - Low: Uniform orientation (smooth surface)
  - High: Variable orientation (edges, rough surfaces)