Promising Methods to Handling Numerical Ties in PCPs

The paper outlines six methods to resolve numerical ties in datasets, enhancing data visualization and analysis. **Attempt 1: Incremental Offset** adds a small value to tied figures based on their placement within a group to maintain the original order. **Attempt 2: Jitter Spacing** randomly modifies tied values to break the ties while preserving the overall structure. **Attempt 3: Random Rank Shuffle** assigns random ranks to tied values, effectively functioning for both numbers and categories. **Attempt 4: Cascade Rank Tie-Breaker** utilizes previous rankings to resolve ties, ensuring the order is clear and comprehensible. **Attempt 5: Tie Breaking the Band** applies small offsets to distinguish tied values while retaining the original sequence. **Attempt 6: Tie Breaking the Band with Auto Fraction** involves automatic detection of data types, robust handling of unusual data values, and adjustments based on the data's distribution. Each method includes code examples that elucidate its advantages and disadvantages for various data types.

Determine the numerical ties in the dataset

Let's determine the number of numerical ties in the dataset.

cat_name	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	year
Tie Counts	170.0	254.0	279.0	240.0	330.0
Tie Percentages	51.1	76.3	83.8	72.1	99.1

–>

–>

Attempt 5: Tie Breaking the Band (Resolving Numerical Ties with Incremental Offset)

Equation:

The incremental offset technique for numerical tie-breaking is defined as:

$$v_i' = v_i + \Delta * k$$

Where:

v_i': Adjusted value for the i-th data point. v_i: Original value for the i-th data point. Δ: A small incremental offset (e.g., 0.01). k: The relative position of the value within the tie group.

Advantages:

- Maintains Sequential Order: Ensures that the original order of tied values is preserved.
- Simple and Deterministic: The adjustment is straightforward and avoids randomness.
- Improves Visualization Clarity: Slight differences between tied values enhance the interpretability of parallel coordinate plots (PCPs).
- Scalable: Handles large datasets efficiently, provided ties are reasonably distributed.

Disadvantages:

- Fixed Offset May Bias Results: Applying a uniform incremental offset could introduce a small, consistent bias in downstream analyses.

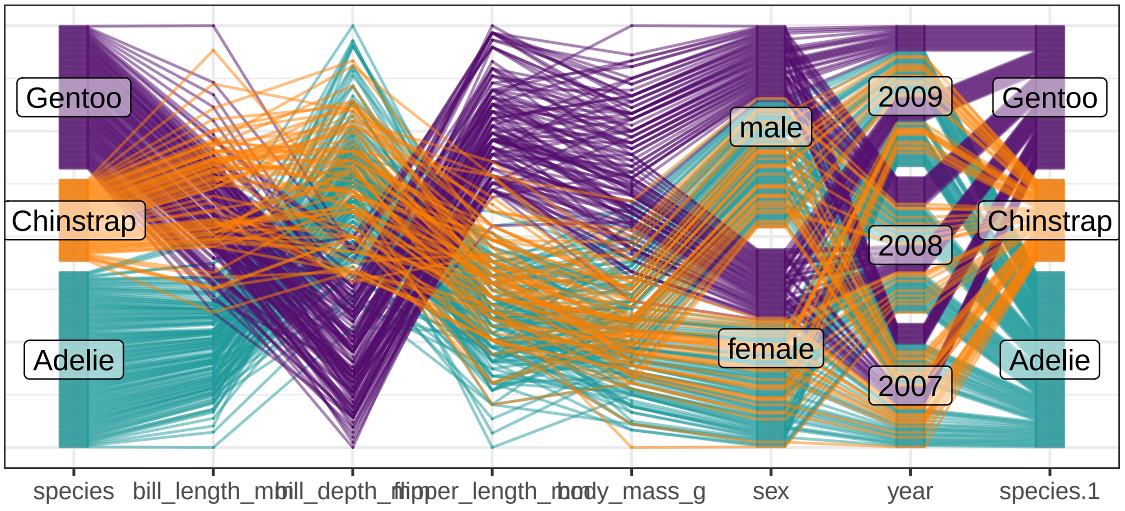
- Less Effective for Dense Ties: When ties are highly frequent, offsets might still result in visual overlap or distortions in high-resolution plots.
- Sequential Dependency: Adjustments depend on the sorted order of the data, which may not be ideal for certain applications.

Implementation:

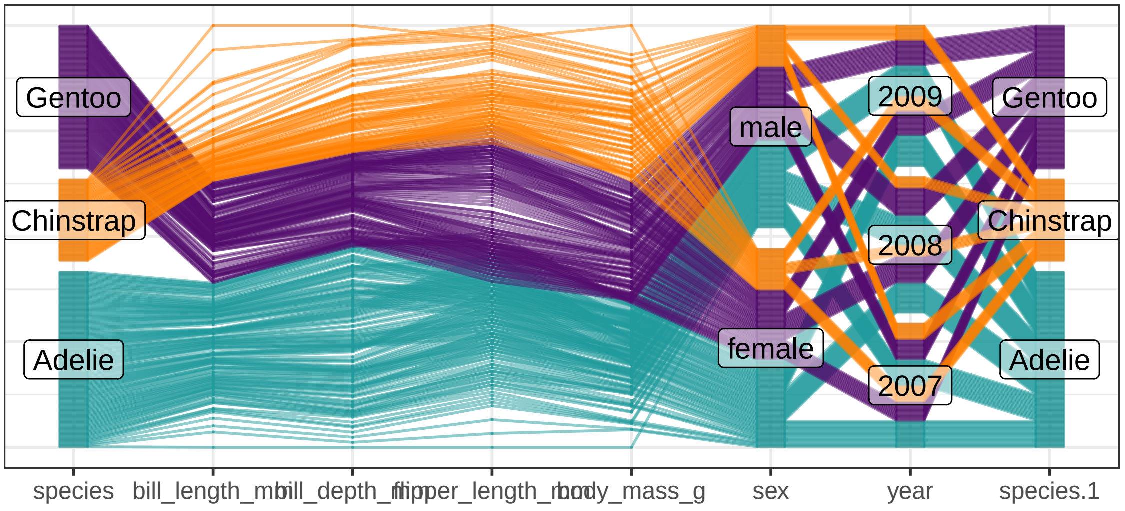
```r
# This function takes a numerical vector and adds a small band around ties
numerical_tie_breaker <- function(values, tie_band = 0.1) {
  # Sort the values
  sorted_values <- sort(values)

  # Initialize the adjusted values
  adjusted_values <- sorted_values

  # Loop through the values to add tie band for ties
  for (i in seq_along(sorted_values)) {
    if (i > 1 && sorted_values[i] == sorted_values[i - 1]) {
      adjusted_values[i] <- adjusted_values[i - 1] + tie_band
    }
  }

  # Return the adjusted values
  return(adjusted_values)
}
```

PCP Implementation

Unadjusted PCP of Palmer Penguins



Incremental Tie−Breaking PCP of Palmer Penguins



Attempt 6: Tie Breaking the Band with auto fraction (Resolving Numerical Ties with Incremental Offset)

Equation:

Step 1 - Detect data type:

if $\forall x \in \mathbb{Z} \implies$ use a small fixed increment (e.g., α).

else $\implies$ continue to outlier check.

Step 2 - Outlier check ($1.5 \times IQR$ **rule):**

$IQR = Q3 - Q1$; $LowerBound = Q1 - 1.5 \times IQR$; $UpperBound = Q3 + 1.5 \times IQR$

$has_outliers = (min(values) < LowerBound) \vee (max(values) > UpperBound)$

Step 3 - Determine tie band (`tie_band`):

$$tieband = \begin{cases} \text{if no outliers:} & \alpha \text{ x SD (values)} \\ \text{if outliers:} & \alpha \text{ x IQR (values)} \\ \text{if whole-number:} & \alpha \end{cases}$$

Step 4 - Increment duplicate sorted values by `tie_band` whenever ties are detected:

The incremental offset technique for numerical tie-breaking is defined as:

$$v_i' = v_i + \Delta * k$$

Where:

v'_i: Adjusted value for the i-th data point. v_i: Original value for the i-th data point. Δ: A small incremental offset (e.g., 0.01). k: The relative position of the value within the tie group.

Advantages:

- Automated data-type detection: Distinguishes whole-number from continuous data.
- Normality check: Uses Shapiro-Wilk to detect Gaussian vs. non-Gaussian data (if the sample size permits).
- Robust outlier consideration: Employs $1.5 \times IQR$ to switch to a robust measure of spread (IQR) when outliers exist or when data are non-Gaussian.

Disadvantages:

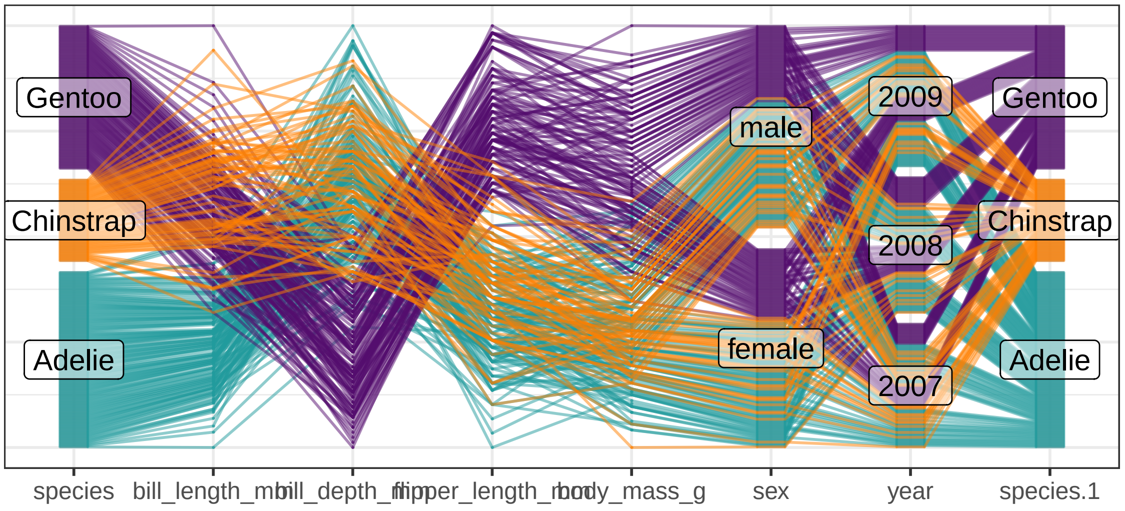
- Reliance on Shapiro-Wilk: The test can be sensitive for large samples and is invalid for very small samples (n < 3).
- Cutoff choices are somewhat arbitrary: The 0.05 alpha in Shapiro-Wilk and 1.5×IQR threshold are common but not universal.
- Potential instability for very small datasets: Both outlier detection and normality testing can be less reliable with limited data.

Implementation:

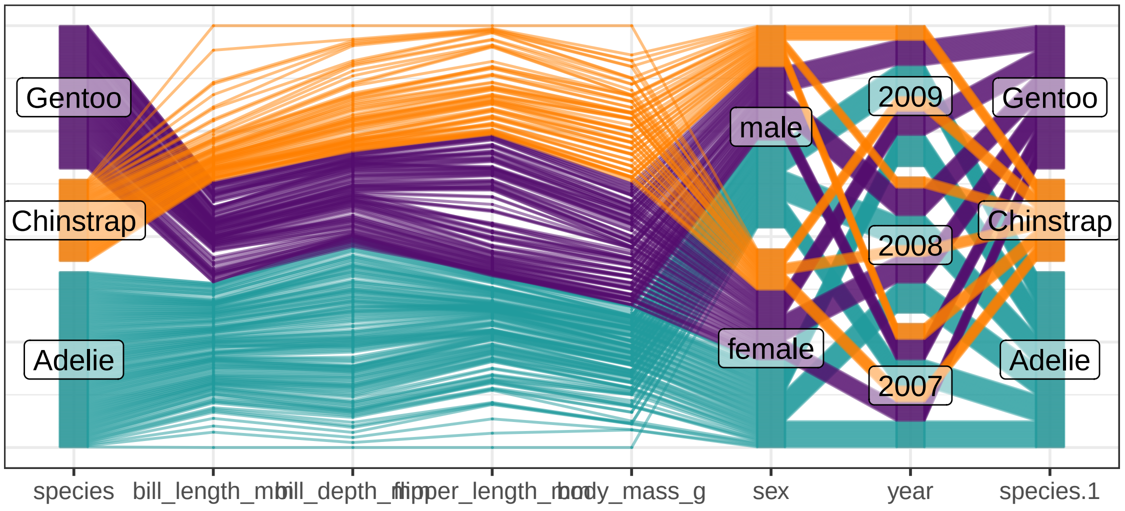
```r
auto_fraction <- function(values, scale_factor = 0.1) {
  unique_vals <- unique(sort(values))
  diffs <- diff(unique_vals)

  # Keep only positive gaps
  positive_diffs <- diffs[diffs > 0]

  # If everything is identical or only one unique value, return a small default
  if (length(positive_diffs) == 0) {
    return(scale_factor)
  }

  # A robust choice: median of these positive gaps
  typical_gap <- median(positive_diffs)

  # Return a fraction that is a multiple of the median gap
  scale_factor * typical_gap
}

numerical_tie_breaker <- function(values, base_fraction = 0.1) {
  # 1) Determine if all values are whole numbers
  is_whole_number <- all(abs(values - round(values)) < .Machine$double.eps^0.5)

  # 2) If data are all whole numbers, use a small fraction increment
  if (is_whole_number) {
    tie_band <- base_fraction

  } else {
    # For continuous data, attempt a normality check using Shapiro-Wilk if n >= 3
    # (Shapiro test not valid for n < 3, so skip if too small)
    is_normal <- FALSE
    if (length(values) >= 3) {
      shapiro_p <- shapiro.test(values)$p.value
      is_normal <- (shapiro_p > 0.05)
    }

    # Check for outliers using 1.5 * IQR rule
    q <- stats::quantile(values, probs = c(0.25, 0.75))
    iqr_value <- q[2] - q[1]
    lower_bound <- q[1] - 1.5 * iqr_value
    upper_bound <- q[2] + 1.5 * iqr_value
    has_outliers <- any(values < lower_bound) || any(values > upper_bound)

    # Decide tie_band logic
    # If data appear normal and have no outliers, use fraction * SD; otherwise use fraction * IQR
    if (is_normal && !has_outliers) {
      tie_band <- base_fraction * stats::sd(values)
    } else {
      tie_band <- base_fraction * iqr_value
    }
  }

  # Optionally, derive the fraction from actual gaps in data (robust approach)
  # fraction <- auto_fraction(values, base_fraction)
  # tie_band <- fraction * <sd or iqr or fixed increment>

  # 3) Sort the values
  sorted_values <- sort(values)

  # 4) Adjust for ties by adding tie_band to each duplicate
  adjusted_values <- sorted_values
  for (i in seq_along(sorted_values)) {
    if (i > 1 && sorted_values[i] == sorted_values[i - 1]) {
      adjusted_values[i] <- adjusted_values[i - 1] + tie_band
    }
  }

  return(adjusted_values)
}
```

PCP Implementation

Unadjusted PCP of Palmer Penguins



Auto Fraction with Incremental Tie−Breaking PCP of Palmer Penguins


Gentoo
Chinstrap
Adelie
male
female
Gentoo
Chinstrap
Adelie
species
bill_length_m
bill_depth_m
flipper_length_m
body_mass_g
sex
year
species.1