Extension of the Dip-test Repertoire - Efficient and Differentiable p-value Calculation for Clustering

SIAM International Conference on Data Mining (SDM23) 
April 27-29, 2023, Minneapolis, Minnesota, U.S.

Lena G. M. Bauer*,1, Collin Leiber*,2, Christian Böhm1, Claudia Plant1

1 Universität Wien, Vienna, Austria  
2 Ludwig-Maximilians-Universität, Munich, Germany

* Authors contributed equally
Outline

- The Dip-test and its $p$-value
- Function fit
- $p$-value experiments
- Gradient of the $p$-value
- The Dip’n’Sub algorithm
- Clustering experiments
- Conclusion
The Dip-test and its $p$-value (I)

- Measures the modality in one-dimensional samples
- Returns a Dip-value $D \in (0, 0.25]$
  - $D \approx 0 \rightarrow$ unimodal
  - $0 \ll D \leq 0.25 \rightarrow$ multimodal
- Is calculated by fitting a unimodal distribution onto the Empirical Cumulative Distribution Function (ECDF)
The Dip-test and its $p$-value (II)

- Used by various clustering algorithms to determine the number of clusters

- But: Both values are far from 0.25

- How high must a Dip-value be to indicate a rather multimodal structure?
  → use $p$-value instead

- $D = 0.0096$

- $D = 0.1103$
The Dip-test and its $p$-value (III)

- We can obtain a corresponding $p$-value for each Dip-value
- The transformation from Dip- to $p$-value is dependent on the sample size $N$
- Existing approaches use precomputed bootstrapped look-up tables
- Maximum number of samples in the most common table is 72,000
- **Our goal**: Designing a transformation function to obtain $p$-values for each combination of Dip-value and $N$
Function fit (I)

- We computed 307 (Dip, p)-pairs for 63 sample sizes by sampling from a uniform distribution with 100,000 repetitions.

- To fit our transformation function, we are guided by the generalised logistic function: 
  \[ p(D) = d + \frac{a - d}{(c + h \cdot e^{-b \cdot D})^{1/9}} \]

- Our final result:
  \[ \hat{p}(D, \hat{b}(N)) = 1 - \left[ 0.6 \cdot (1 + 1.6 \cdot e^{-\hat{b}(N) \cdot D + 6.5})^{1/1.6} + 0.4 \cdot (1 + 0.2 \cdot e^{-\hat{b}(N) \cdot D + 6.5})^{1/0.2} \right]^{-1} \]

  with \( \hat{b}(N) = 17.30784 \cdot \sqrt{N} + 12.04918 \)
Function fit (II)

- Our transformation function is well-defined for all $N$ and all Dip-values
- It allows a fast and precise calculation of $p$-values
Function fit (II)

- Our transformation function is well-defined for all $N$ and all Dip-values
- It allows a fast and precise calculation of $p$-values

Regarding the computed (Dip-p)-pairs we achieve a MSE of $3.43\times10^{-6}$
→ The original look-up table achieves a MSE of $7.92\times10^{-6}$

Regarding previously unseen data we achieve a MSE of $3.14\times10^{-6}$
→ The original look-up table achieves a MSE of $8.12\times10^{-6}$
### p-value experiments

- **p-values for different unimodal (left) and multimodal (right) distributions**
- **All given values are averages for 100 random samples**
- **T = Look-up table, F = Function fit, B = Bootstrapped**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{N}(4,1)$</td>
<td>T</td>
<td>0.77 ± 0.24</td>
<td>0.86 ± 0.19*</td>
<td>0.97 ± 0.07*</td>
<td>†</td>
<td>$\mathcal{N}(4,1)$</td>
<td>8.94 ± 15.0</td>
<td>0.06 ± 0.23*</td>
<td>0.00 ± 0.00*</td>
<td>†</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>0.77 ± 0.24</td>
<td>0.86 ± 0.19</td>
<td>0.97 ± 0.07</td>
<td>1.00 ± 0.02</td>
<td>$\mathcal{U}$</td>
<td>8.83 ± 14.9</td>
<td>0.09 ± 0.25</td>
<td>0.00 ± 0.00</td>
<td>0.00 ± 0.00</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>0.77 ± 0.24</td>
<td>0.86 ± 0.19</td>
<td>0.97 ± 0.07</td>
<td>1.00 ± 0.02</td>
<td>$\mathcal{N}(0,1)$</td>
<td>8.78 ± 15.0</td>
<td>0.06 ± 0.21</td>
<td>0.00 ± 0.00</td>
<td>0.00 ± 0.00</td>
</tr>
<tr>
<td>$\mathcal{T}_{nc}(4,2,0.1)$</td>
<td>T</td>
<td>0.80 ± 0.21</td>
<td>0.89 ± 0.14*</td>
<td>0.98 ± 0.03*</td>
<td>†</td>
<td>$\mathcal{T}_{nc}(4,2,0.1)$</td>
<td>0.79 ± 2.20</td>
<td>0.00 ± 0.00*</td>
<td>0.00 ± 0.00*</td>
<td>†</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>0.80 ± 0.21</td>
<td>0.90 ± 0.14</td>
<td>0.98 ± 0.03</td>
<td>1.00 ± 0.00</td>
<td>$\mathcal{U}$</td>
<td>0.83 ± 2.07</td>
<td>0.00 ± 0.00</td>
<td>0.00 ± 0.00</td>
<td>0.00 ± 0.00</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>0.80 ± 0.21</td>
<td>0.90 ± 0.14</td>
<td>0.99 ± 0.03</td>
<td>1.00 ± 0.00</td>
<td>$\mathcal{T}_{nc}(4,2,7.1)$</td>
<td>0.74 ± 2.17</td>
<td>0.00 ± 0.00</td>
<td>0.00 ± 0.00</td>
<td>0.00 ± 0.00</td>
</tr>
<tr>
<td>$\mathcal{L}(0,2)$</td>
<td>T</td>
<td>0.85 ± 0.19</td>
<td>0.95 ± 0.11*</td>
<td>0.99 ± 0.04*</td>
<td>†</td>
<td>$\mathcal{L}(0,2)$</td>
<td>24.1 ± 24.9</td>
<td>2.36 ± 7.87*</td>
<td>0.00 ± 0.00*</td>
<td>†</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>0.85 ± 0.19</td>
<td>0.95 ± 0.11</td>
<td>0.99 ± 0.04</td>
<td>1.00 ± 0.00</td>
<td>$\mathcal{U}$</td>
<td>23.9 ± 25.1</td>
<td>2.37 ± 7.82</td>
<td>0.00 ± 0.00</td>
<td>0.00 ± 0.00</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>0.85 ± 0.19</td>
<td>0.95 ± 0.11</td>
<td>0.99 ± 0.04</td>
<td>1.00 ± 0.00</td>
<td>$\mathcal{L}(7,2)$</td>
<td>23.8 ± 24.8</td>
<td>2.34 ± 8.06</td>
<td>0.00 ± 0.00</td>
<td>0.00 ± 0.00</td>
</tr>
</tbody>
</table>

Multiplied by 100
Gradient of the $p$-value

- Another advantage is the differentiability of our function

$$
\nabla_\rho (\hat{p}(D, \hat{b}(N))) = (-\hat{b}(N) \nabla_\rho (D)) \cdot e^{-\hat{b}(N) \cdot D + 6.5} \cdot \left[ 0.6 \cdot (1 + 1.6e^{\hat{b}(N) \cdot D + 6.5})^{-\frac{6}{1.6}} + 0.4 \cdot (1 + 0.2e^{\hat{b}(N) \cdot D + 6.5})^{\frac{8}{0.2}} \right]^{-2}
$$

- This gradient can be utilized to identify cluster-friendly projection axes using stochastic gradient descent
- Those axes can form a lower-dimensional subspace of the given dataset
- In theory this gradient can also be used for Deep Learning applications
The Dip’n’Sub algorithm (I)

- Simultaneously defines the clusters and a matching subspace without knowing the number of clusters $k$ in advance
- Iteratively identifies a new projection axis $\rho$ that minimizes:
  \[
  \frac{1}{N} \sum_{i=1}^{k} |C_i| \hat{p}(\text{Dip}(\overline{C_i}^{\rho}), \hat{b}(|C_i|))
  \]
- Check if more than $T\%$ of samples are contained in a multimodal cluster on $\rho$
- Split clusters on $\rho$ using our subroutine \textit{TailoredDip}
The Dip’n’Sub algorithm (I)

- Simultaneously defines the clusters and a matching subspace without knowing the number of clusters $k$ in advance
- Iteratively identifies a new projection axis $\rho$ that minimizes:
  \[
  \frac{1}{N} \sum_{i=1}^{k} |C_i| \hat{p}(\text{Dip}(\overline{C_i^\rho}), \hat{b}(|C_i|))
  \]
- Check if more than $T\%$ of samples are contained in a multimodal cluster on $\rho$
- Split clusters on $\rho$ using our subroutine \textit{TailoredDip}
The Dip’n’Sub algorithm (II)

Input dataset
The Dip’n’Sub algorithm (II)

Input dataset

1st identified axis
The Dip’n’Sub algorithm (II)

Input dataset

1st identified axis

2nd identified axis
The Dip’n’Sub algorithm (II)

Input dataset

1st identified axis

Final result

2nd identified axis
Clustering experiments (I)

- Maximum NMI results of different common subspace and Dip-based $k$-estimation algorithms after 10 runs

<table>
<thead>
<tr>
<th>Dataset ($k/d$)</th>
<th>Common Subspace Algorithms</th>
<th>Dip-based $k$-estimation Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYNTH (7/8)</td>
<td>0.97 (7/2)</td>
<td>0.87 (7/4)</td>
</tr>
<tr>
<td>BANK (2/4)</td>
<td>0.41 (7/3)</td>
<td>0.03 (2/2)</td>
</tr>
<tr>
<td>USER (4/5)</td>
<td>0.52 (10/1)</td>
<td>0.43 (4/5)</td>
</tr>
<tr>
<td>HTRU2 (2/8)</td>
<td>0.38 (3/2)</td>
<td>0.03 (2/2)</td>
</tr>
<tr>
<td>ALOI (4/66)</td>
<td>0.98 (4/1)</td>
<td><strong>1.00</strong> (4/35)</td>
</tr>
<tr>
<td>MICE (8/68)</td>
<td>0.55 (6/3)</td>
<td>0.27 (8/6)</td>
</tr>
<tr>
<td>AIBO (2/70)</td>
<td><strong>0.68</strong> (2/1)</td>
<td><strong>0.68</strong> (2/20)</td>
</tr>
<tr>
<td>MOTE (2/84)</td>
<td><strong>0.37</strong> (2/1)</td>
<td>0.30 (2/42)</td>
</tr>
<tr>
<td>SYMB (6/398)</td>
<td>0.84 (5/3)</td>
<td>0.80 (6/6)</td>
</tr>
<tr>
<td>OLIVE (4/570)</td>
<td>0.57 (4/2)</td>
<td>0.68 (4/4)</td>
</tr>
</tbody>
</table>
Clustering experiments (II)

<table>
<thead>
<tr>
<th>Dataset ((k/d))</th>
<th>Common Subspace Algorithms</th>
<th>Dip-based (k)-estimation Algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYMB ((6/398))</td>
<td>0.84 ((5/3))</td>
<td>0.80 ((6/6))</td>
</tr>
</tbody>
</table>

- Dip’n’Sub (ACC=0.79)
- PCA+KM (ACC=0.69)
- ICA+KM (ACC=0.71)
- DipExt+KM (ACC=0.57)
- LDA-KM (ACC=0.69)
- SubKM (ACC=0.69)
Conclusion

- We created a differentiable function to translate Dip- to $p$-values → Automatic and fast transformation for any sample size
- Its error regarding MSE is lower than previously used look-up tables
- The gradient of our function can be useful for the Data Mining community as shown by our Dip’n’Sub clustering algorithm
- Experiments show that Dip’n’Sub produces high-quality clustering results while simultaneously defining a lower-dimensional subspace

Thank you for your attention!

Contact: leiber@dbs.ifi.lmu.de
Algorithm

**Algorithm 1: The Dip’n’Sub algorithm**

**Input:** data set $X$, significance $\alpha$, threshold $T$

**Output:** labels

1. $k = 1$; labels = $[0,\ldots,0]$; $X_{fin} = []$

2. while True do
3.   $s = 1$; $\rho = 0$
4.   $Q = \log(d)$ features with lowest weighted p-values $\cup$ first $\log(d)$ components of PCA
5.   for each $\rho_{tmp} \in Q$ do
6.     Update $\rho_{tmp}$ with SGD using Eq. 3.1
7.     $s_{tmp} = \text{value of Eq. 3.1 using } \rho_{tmp}$
8.     if $s_{tmp} < s$ then
9.       $s = s_{tmp}$; $\rho = \rho_{tmp}$
10. $P = \{\text{p-value(Dip}(C_i^p), |C_i^p|) \mid i \in [1,k]\}$
11. if $\sum_{i=1}^{k} |C_i| \geq T$ then
12.   for each cluster $i$ with $P_i < \alpha$ do
13.     $\text{labels}_{new} = \text{TailoredDip}(C_i^p, \alpha)$
14.     update labels using $\text{labels}_{new}$
15.     $X_{fin} = \text{combine } X_{fin} \text{ and } \{\rho^T x \mid x \in X\}$
16.     $X = \text{keep features orthogonal to } \rho$
17.   else
18.     break
19. return labels, $X_{fin}$

Eq. 3.1:

$$\frac{1}{N} \sum_{i=1}^{k} |C_i| \hat{p}(\text{Dip}(\overline{C_i^p}), \hat{b}(|C_i|))$$
**p-value experiments using clustering algorithms**

- Average NMI and runtime (RT) results for different Dip-based clustering methods after 10 runs

| Dataset | | DipMeans | | p. DipMeans | | SkinnyDip |
|---------|----------------------|----------------------|----------------------|----------------------|----------------------|
|         | NMI | NMI | RT | T | F | B | T | F | B | T | F | B | T | F | B | T | F | B | T | F | B | T | F | B |
| SYNTH   | 0.64 | 0.64 | 0.64 | 6.87 | 5.82 | 8.92 | 0.86 | 0.86 | 0.85 | 0.76 | 0.74 | 9.03 | 0.16 | 0.16 | 0.16 | 0.02 | 0.01 | 14.66 |
| BANK    | 0.31 | 0.31 | 0.30 | 13.00 | 6.93 | 43.71 | 0.30 | 0.31 | 0.30 | 7.70 | 5.46 | 69.81 | 0.13 | 0.13 | 0.13 | 0.01 | 0.00 | 2.56 |
| USER    | 0.00 | 0.00 | 0.00 | 0.05 | 0.02 | 0.08 | 0.34 | 0.34 | 0.35 | 1.54 | 1.13 | 11.37 | 0.15 | 0.15 | 0.15 | 0.00 | 0.00 | 0.68 |
| HTRU2   | 0.00 | 0.00 | 0.00 | 28.87 | 28.32 | 29.03 | 0.17 | 0.17 | 0.17 | 2.20 | 2.18 | 19.05 | 0.08 | 0.08 | 0.08 | 0.04 | 0.04 | 40.78 |
| ALOI    | 0.96 | 0.96 | 0.92 | 0.43 | 0.22 | 0.84 | 0.49 | 0.51 | 0.49 | 9.03 | 6.68 | 54.36 | 0.17 | 0.15 | 0.17 | 0.06 | 0.04 | 3.68 |
| MICE    | 0.00 | 0.00 | 0.00 | 0.19 | 0.12 | 0.27 | 0.53 | 0.54 | 0.53 | 91.47 | 50.21 | 607.21 | 0.00 | 0.00 | 0.00 | 0.01 | 0.01 | 3.76 |
| AIBO    | 0.00 | 0.00 | 0.00 | 0.09 | 0.04 | 0.14 | 0.28 | 0.33 | 0.27 | 20.02 | 5.33 | 129.60 | 0.02 | 0.02 | 0.02 | 0.02 | 0.01 | 3.29 |
| MOTE    | 0.35 | 0.35 | 0.35 | 0.85 | 0.62 | 1.21 | 0.00 | 0.00 | 0.00 | 0.03 | 0.02 | 0.37 | 0.00 | 0.00 | 0.00 | 0.03 | 0.02 | 8.73 |
| SYMB    | 0.82 | 0.82 | 0.82 | 1.50 | 1.15 | 2.23 | 0.70 | 0.74 | 0.70 | 36.70 | 5.64 | 73.69 | 0.02 | 0.02 | 0.02 | 0.11 | 0.10 | 5.16 |
| OLIVE   | 0.50 | 0.50 | 0.50 | 0.06 | 0.04 | 0.16 | 0.64 | 0.52 | 0.64 | 1.26 | 0.15 | 3.08 | 0.04 | 0.04 | 0.04 | 0.03 | 0.02 | 0.39 |
## Dataset information

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$N$</th>
<th>$d$</th>
<th>$k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYNTH</td>
<td>6,300</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>BANK</td>
<td>1,372</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>USER</td>
<td>403</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>HTRU2</td>
<td>17,898</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>ALOI</td>
<td>288</td>
<td>66</td>
<td>4</td>
</tr>
<tr>
<td>MICE</td>
<td>1,077</td>
<td>68</td>
<td>8</td>
</tr>
<tr>
<td>AIBO</td>
<td>621</td>
<td>70</td>
<td>2</td>
</tr>
<tr>
<td>MOTE</td>
<td>1,272</td>
<td>84</td>
<td>2</td>
</tr>
<tr>
<td>SYMB</td>
<td>1,020</td>
<td>398</td>
<td>6</td>
</tr>
<tr>
<td>OLIVE</td>
<td>60</td>
<td>570</td>
<td>4</td>
</tr>
</tbody>
</table>
Runtime experiments

The graph shows the time (in logs) taken by different methods (bootstrap, table, function) as a function of N (equally spaced scale). The time increases significantly with increasing N for the bootstrap method, while the other methods show more stable performance.