# EFFICIENT COMPUTATION OF K-SUBSET SUM DISTRIBUTIONS VIA TRANSLATION INVARIANCE

January 5, 2026

## 1 Methodology

### 1.1 Analysis of $k$-Group Distributions Discrete Derivatives

Let $\mathbf{f}^{k,n}$ denote the frequency vector of sums of $k$-element subsets of $A$, without repetitions. To better understand the behaviour of the distributions of sums of $k$-element subsets ($k$-group distributions) as $k$ and $n$ vary, we perform an experimental analysis by applying iterative discrete differentiation to the frequency vectors $\mathbf{f}^{k,n}$. This technique reveals underlying patterns and structures within the distributions.

The discrete differentiation process involves subtracting adjacent elements of the frequency vector iteratively. For a frequency vector $\mathbf{f} = (f_1, f_2, \ldots, f_m)$, the first discrete derivative is defined as:

$$\Delta\mathbf{f} = (\Delta f_1, \Delta f_2, \ldots, \Delta f_{m-1}), \quad \text{where } \Delta f_i = f_{i+1} - f_i.$$

This process is generalized for higher-order derivatives by repeatedly applying the differentiation operation. For $k$-th order differentiation, we define:

$$\Delta^k\mathbf{f} = \Delta(\Delta^{k-1}\mathbf{f}),$$

where $\Delta^0\mathbf{f} = \mathbf{f}$.

In our case, however, we adopt a variant of discrete differentiation that incorporates growing offsets. Specifically, for each iteration $k_{\text{inner}}$, we subtract elements at increasing distances from the current position. Formally, let the vector after $k_{\text{inner}}-$th differentiation be $\mathbf{d} = (d_1, d_2, \ldots, d_r)$. The next iteration is computed as:

$$d'_i = d_i - d_{i-k_{\text{inner}}}, \quad \text{for all } i \geq k_{\text{inner}}.$$

### 1.2 Patterns in $k$-Group Distributions through Iterative Differentiation

Our experimental analysis reveals a regularity in the behaviour of $k$-group distributions when subjected to $k$-th order discrete differentiation. Specifically, for any fixed $k$, the resulting differentiated distribution exhibits a consistent pattern across all $n$, differing only by a translation.

**Observed Pattern** When iteratively differentiating the frequency vector $\mathbf{f}^{k,n}$ $k$-times as described in the previous section, we observe the following:

- The $k$-th order derivative vanishes in all positions except at specific points corresponding to the positions of bell-shaped distributions of the previous $k-1$ layers.
- These peaks represent the contributions of earlier $(k-1)$-group distributions to the current $k$-group distribution.
- The differentiated distribution for any $n$ is therefore identical in shape to the pattern observed for smaller $n$, shifted appropriately to align with the positions of the peaks.

**Translation Invariance** For any $n$, the differentiated distribution is invariant up to a translation determined by the position of the peaks. This implies that the total distribution can be reconstructed deterministically by knowing:

1. The positions of the peaks, which correspond to the contributions of the previous $(k-1)$-group distributions.
2. The shape of the distribution, which remains unchanged across $n$.

Figure 1 illustrates this behavior for various values of $n$ and a fixed $k$, showing how the differentiated distributions are identical in shape but translated depending on $n$.

In Figure 2, we instead focus on the peaks of the same distributions plotted earlier, zoomed for $k = 8$. These peaks coincide with the distributions of the previous $k-1$-group subsets, as plotted in the second row for $n = 8$. This demonstrates that the recursive relationship between distributions not only preserves the pattern but also aligns the peaks with contributions from previous distributions, highlighting the structural invariance across different $k$ and $n$.

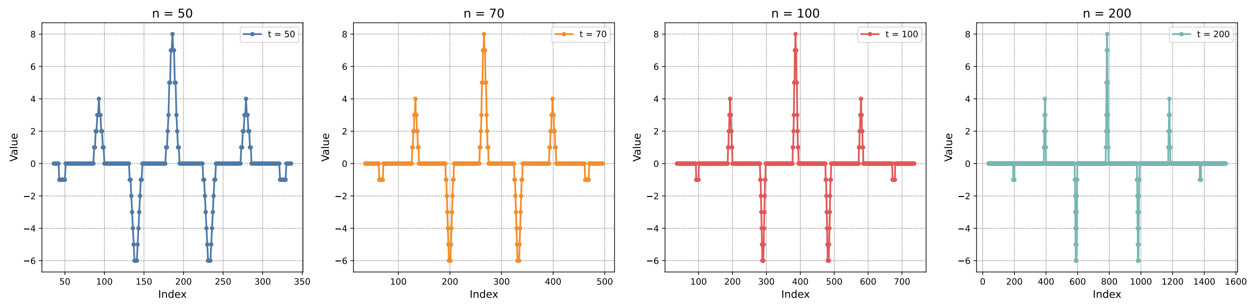

Figure 1: Differentiated distributions of $k$-group subsets for a fixed $k$ and varying $n$ values ($n = 100, 200, 300, 400$). The figure demonstrates how the differentiated distributions exhibit identical patterns across different $n$, differing only by a translation.

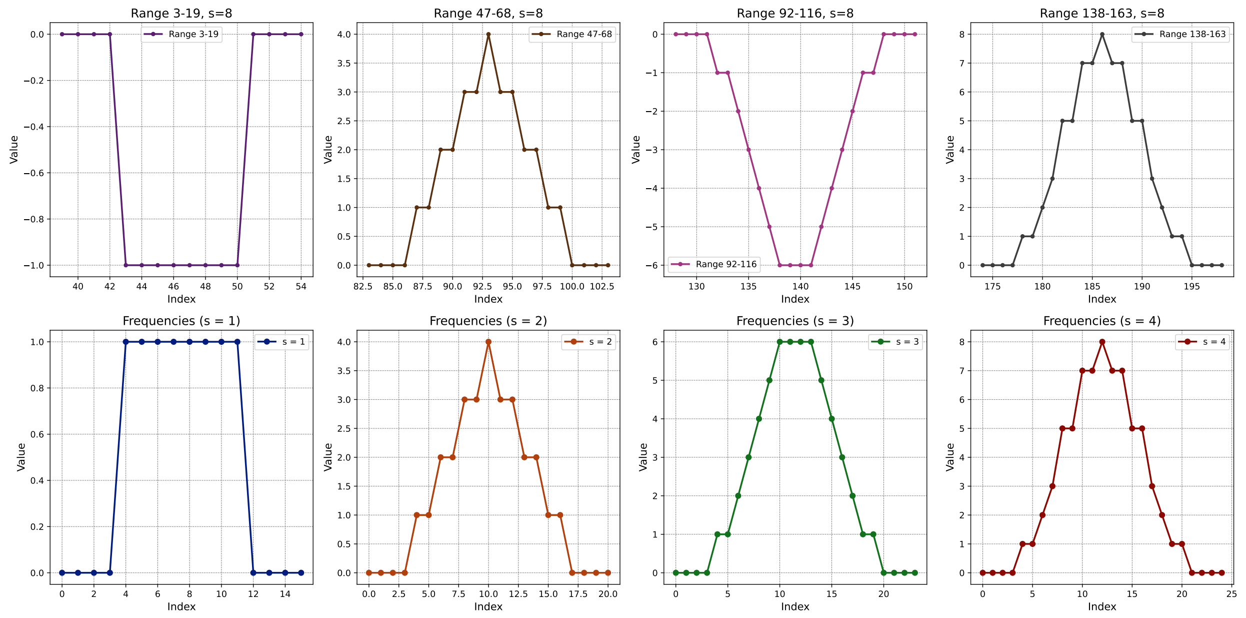

Figure 2: Zoomed view of the differentiated distributions for $k$-group subsets with $k = 8$. The peaks observed in the first row align precisely with the contributions of the previous $k-1$ group distributions, as shown in the second row for $n = 8$. This demonstrates the recursive relationship between distributions, highlighting how the peaks of the $k$-th group distributions coincide with the bell-shaped structures of the previous groups, thereby reinforcing the structural invariance across $k$ and $n$.

**Patterns for Peaks** We now investigate the precise nature of the peaks observed in the differentiated distribution of $k$-group subsets. Experimental results, as shown in Figure 2, reveal that for $k = 8$, there are a total of seven peaks. Due

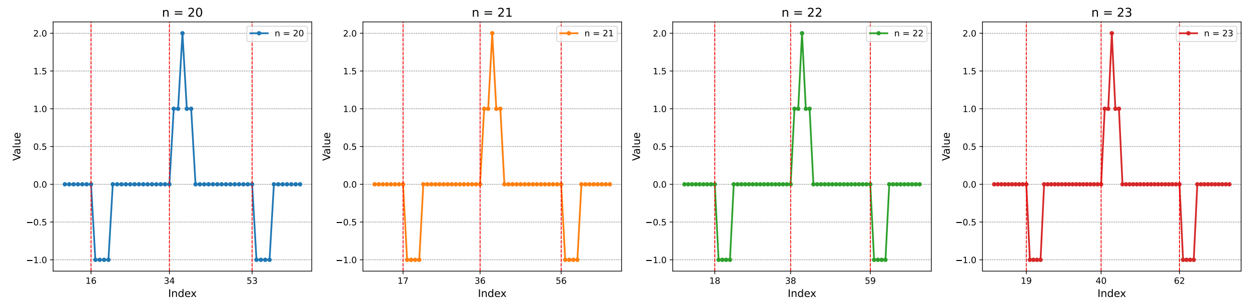

Figure 3: Progression of the translation pattern for $k = 4$ and $n = 20, 21, 22, 23$. The red vertical lines mark the positions of the peaks, which increment by $+1, +2, +3$ at each step, corresponding to the translation rule for sub-distributions in the hierarchy. This demonstrates the consistency of the translation pattern across increasing $n$.

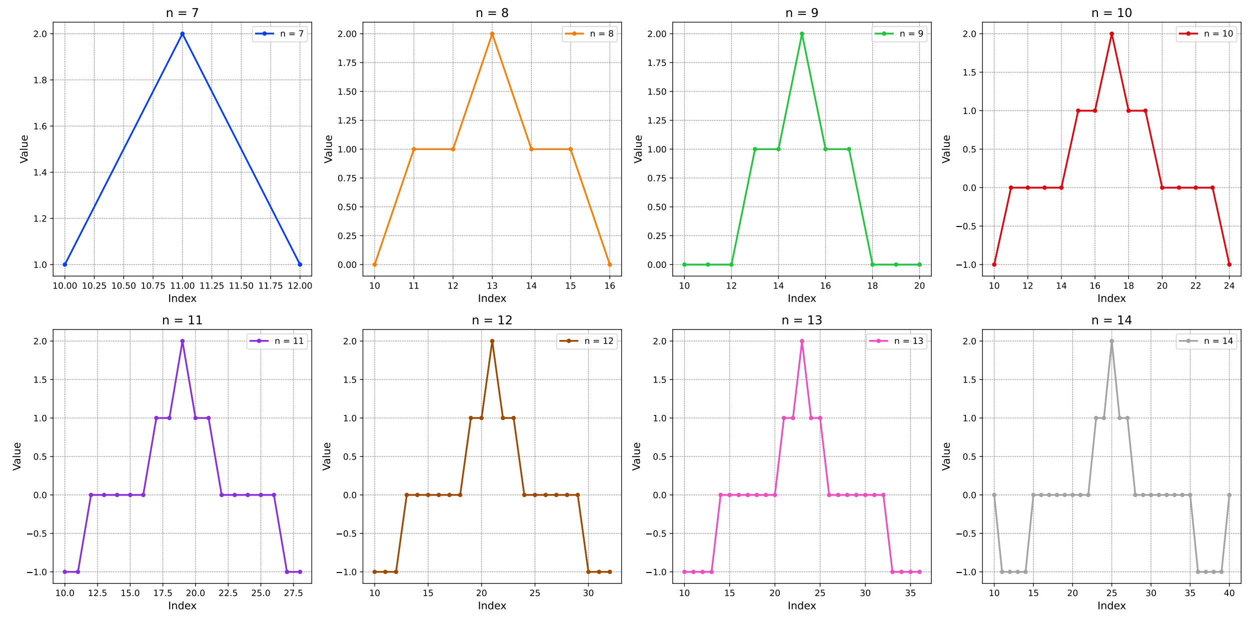

Figure 4: Visualization of the transient phase for $n = 7, 8, 9, 10, 11, 12, 13, 14$. Each subplot represents the differentiated distribution for a specific $n$. During the transient phase, irregularities in the peak structure and alignment can be observed, particularly for smaller values of $n$. As $n$ increases, the distributions gradually stabilize, converging towards the steady translation pattern. This figure highlights the progressive buildup of the final structure and the transition from irregular behavior to consistent alignment across sub-distributions.

to symmetry, it is sufficient to analyze only the first four peaks. These peaks correspond exactly to the distributions of 1-, 2-, 3-, and 4-element groups for $n = 0$ to $8 - 1$.

This pattern is consistent for any $k$: the peaks in the differentiated distribution always align with the distributions of subsets of size $k = 1$ up to $\lfloor k/2 \rfloor$, evaluated over the range $0$ to $k - 1$.

**Pattern for Translation** The patterns governing translation are slightly more complex to analyze. This complexity arises from an initial transitional phase, during which the final steady structure is not yet fully formed but incrementally builds over time. Once this transitional phase is complete, the translation pattern becomes straightforward: each sub-distribution present in the differentiation, corresponding to $k - i$, shifts forward by exactly $t$ steps for each increment of $n$, where $t$ is the position of the sub-distribution in the hierarchy. For example, if there are three sub-distributions, the first one shifts by 1 step, the second by 2 steps, and the third by 3 steps with each increment of $n$.

In Figure 3, we observe this progression for $k = 4$ and $n = 20, 21, 22, 23$. There are three peaks, as predicted by the peak pattern rule, and their positions increment precisely by $+1, +2, +3$ at each step. The red vertical lines highlight the positions of these peaks, providing a clear visualization of the regular translation pattern across different values of $n$.

**Transient Phase** Before the steady translation pattern emerges, the system undergoes an initial transient phase during which the structure of the differentiated distribution gradually stabilizes. This phase is characterized by irregularities in the alignment of peaks and sub-distributions, which depend on both $k$ and the initial values of $n$.

During this phase, not all sub-distributions are fully formed or aligned according to the final translation rule. Instead, they progressively build up and start contributing to the overall structure. Observing Figure 4, it becomes evident that the distribution develops from the central peak, gradually expanding outward to include additional peaks. For smaller $n$, only the central peak is prominent, while as $n$ increases, new peaks emerge symmetrically on both sides, progressively building the full pattern.

The transient phase length varies with $k$: for smaller values of $k$, the steady pattern emerges more quickly, while for larger $k$, the system requires more iterations of $n$ to reach the steady state. This is due to the greater number of sub-distributions that must align and stabilize. Additionally, during the transient phase, the spacing and positions of peaks do not yet follow the steady translation increments of $+1, +2, +3$, adding complexity to the analysis.

The duration of the transient phase can be determined deterministically. Starting with $k = 3$, where it ends at $n = 6$, the duration increases linearly with $k$: for example, for $k = 4$, the transient phase ends at $n = 6 + 4 = 10$; for $k = 5$, it ends at $n = 10 + 5 = 15$, and so on. This behavior follows an arithmetic progression, where the duration of the transient phase increases by $k$ at each step.

In this study, however, we ignore the transient phase to focus exclusively on the steady-state behavior of the system. Specifically, we analyze the problem in the regime where $n$ is significantly larger than $k$, a scenario where traditional DP approaches fail due to their pseudo-exponential complexity. By bypassing the transient phase, we aim to explore efficient solutions for large-scale instances, which are critical in practical applications.

## 1.3 Algorithm

From the previous observations, we deduced the patterns governing the distributions. The key principle exploited in our algorithm is the invariance of the distribution, except for the translation of peaks present in the differentiation. Since in this study we do not explicitly model the transient phase, we utilize the knowledge of the distribution at the first $n$ after the transient phase to construct the distribution for all subsequent $n$.

The differentiation can be modeled using a simple equation, leveraging the fact that the discrete derivative is zero for all points except at the peaks. This insight allows us to efficiently compute the distribution for larger values of $n$ without explicitly iterating over all subsets.

The algorithm used for symbolic differentiation to model the steady-state distribution is detailed in Algorithm 1. For

**Algorithm 1** Symbolic Differentiation to Model Steady-State Distribution

```
Require: $k$ (number of elements in subsets)
Ensure: Symbolic equation representing the discrete differentiation
1: Define symbolic variable $i$ and symbolic function $x(i)$
2: Initialize the equation as expr $= x(i) - x(i-1)$
3: for shift $= 2$ to $k$ do
4:     Substitute $i \to i -$ shift in expr
5:     Update expr $\leftarrow$ expr $-$ expr$_{\text{shift}}$
6: end for
7: Solve the equation expr $= 0$ for $x(i)$ return Symbolic solution $x(i)$
```

example, for $k = 3$, the resulting equation is:

$$x(i-6) - x(i-5) - x(i-4) + x(i-2) + x(i-1) = 0.$$

Here, we observe that the largest index in the equation $(i - 6)$ corresponds exactly to the end of the transient phase. This demonstrates that the symbolic equation not only models the differentiation process but also encodes the end of the transient phase within its structure.

The total process is structured as follows: for a given $k$, we compute the distribution at $n'$ (the $n$ where the transient phase ends) with a standard model, e.g. DP, and save the distribution of peaks. For any $n > n'$, the distribution is constructed using the differentiation equation, adding contributions based on a supplemental function `get_peaks`, which deterministically returns the translated indices of the peaks and their corresponding values. The pseudocode for the process is provided in Algorithm 2.

**Algorithm 2** Compute Distribution Beyond Transient Phase

```
Require: $k$ (number of elements in subsets), $n'$ (end of transient phase)
Ensure: Distribution for $n > n'$
 1: Compute $n' = 6 + \sum_{i=4}^{k} i$
 2: Compute the symbolic equation for $k$
 3: Compute the distribution at $n'$
 4: Save peak indices and their values from the distribution at $n'$
 5: for each $n > n'$ do
 6:     Initialize the distribution for $n$ with zeros
 7:     Retrieve peak indices and values using get_peaks($k, n, n'$)
 8:     for each index $i$ in n do
 9:         Solve the symbolic equation for i
10:         Add $p$value if the contribution of the peak is $! = 0$ in i
11:     end for
12: end for
```

## 2 Evaluation

In this section, we compare the solution presented in the methodology with the state-of-the-art approach, which uses DP. The DP algorithm works as follows: it iteratively builds the frequencies of sums for subsets of size $k$, leveraging previously computed results to avoid redundant calculations.

In Algorithm 3, we present the pseudocode for the dynamic programming algorithm used for this comparison.

To evaluate the performance of our method against the DP approach, we use runtime efficiency, measured as the total time required to compute the distributions for a given $n$ and $k$.

**Algorithm 3** Dynamic Programming for Sum Frequencies

```
Require: numeri_range (range of numbers), $k$ (number of elements)
Ensure: Frequencies of sums using $k$ elements
 1: Initialize $dp[i]$ as an array of counters for $i = 0$ to $k$
 2: Set $dp[0][0] = 1$ // One way to achieve sum 0 with 0 elements
 3: for each number num in numeri_range do
 4:     for $i = k$ to 1 (in reverse order) do
 5:         for each sum $s$ and count count in $dp[i-1]$ do
 6:             Update $dp[i][s + \text{num}] \leftarrow dp[i][s + \text{num}] + \text{count}$
 7:         end for
 8:     end for
 9: end for
10: Extract unique sums somme_uniche and their frequencies frequenze_somme from $dp[k]$
11: return somme_uniche, frequenze_somme
```

**Results** The distributions obtained with the differentiation method coincide perfectly with the ones obtained with DP. Our method demonstrates superior performance compared to the DP approach, particularly in scenarios where $n \gg k$. While the DP approach suffers from pseudo-exponential complexity as $n$ increases, our method exhibits near-constant runtime for fixed $k$.

Figure 5 evaluates the runtime performance for $k = 6$ and $n$ ranging from 100 to 2000. The results clearly show that the runtime of our method remains stable, whereas the DP approach experiences significant growth as $n$ increases.

Figure 6 further explores the performance of both methods by comparing their runtime for various values of $k$ (from 3 to 7) across the same range of $n$. The solid lines, representing our method, demonstrate consistent performance regardless of $k$, while the dashed lines for DP reveal its inefficiency, with runtime increasing sharply as both $n$ and $k$ grow.

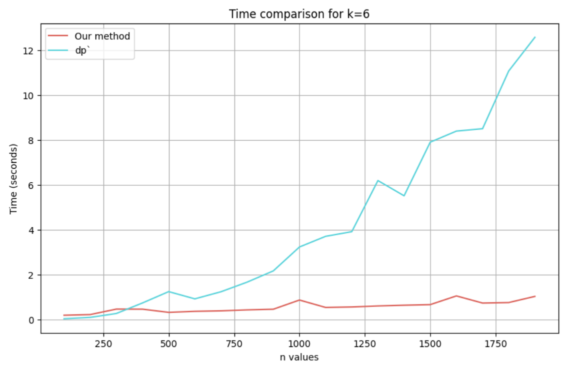

Figure 5: Comparison of runtime between our method and dynamic programming for $k = 6$ and $n$ ranging from 100 to 2000. The plot shows that while the runtime of the DP approach grows rapidly with $n$, our method remains nearly constant across all values of $n$.

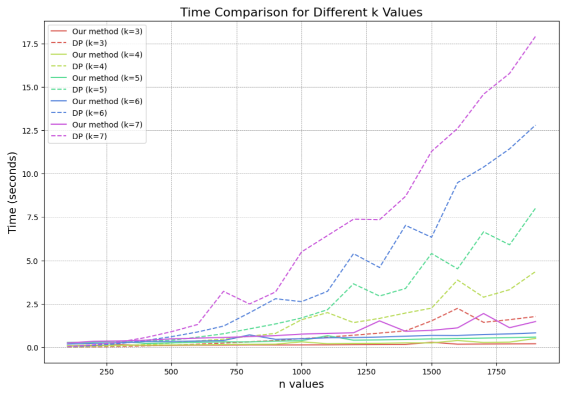

Figure 6: Comparison of runtime scalability for different values of $k$ (from 3 to 7) and $n$ ranging from 100 to 2000. The solid lines represent our method, while the dashed lines correspond to the dynamic programming approach.

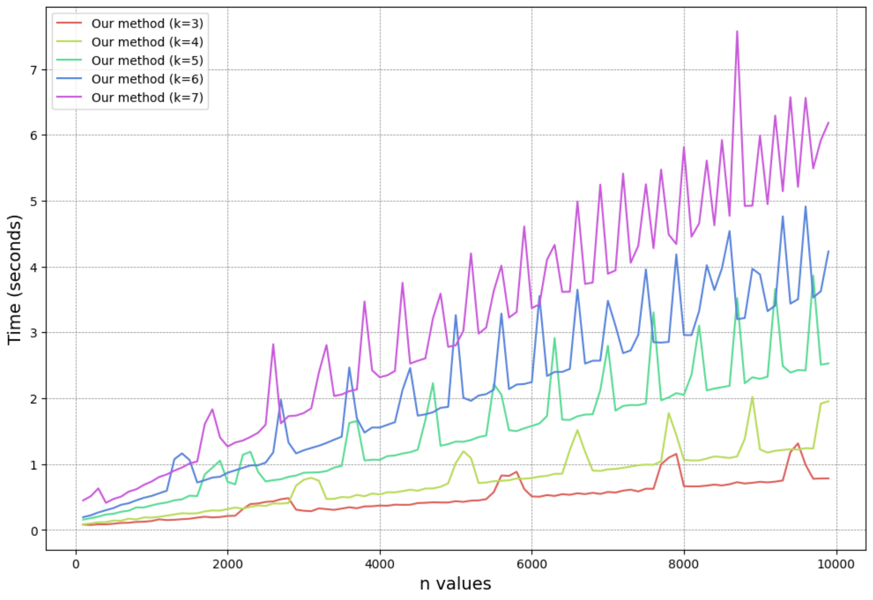

Figure 7: Comparison of execution times for different values of $k$ as $n$ increases. Despite the oscillations, which can be caused by variations in memory access patterns and system-level resource contention, the trend shows that the runtime grows steadily and approximately linearly with larger $n$ and higher values of $k$.

In Figure 7, we extended the evaluation to $n$ values as high as 10,000. The results show that while our method exhibits a steady growth in runtime, the growth is regular and approximately linear, as expected. Notably, even for 10,000 iterations and $k = 7$, the runtime of our method remains under 7 seconds. In comparison, the DP approach required approximately 17.5 seconds for $n = 1,000$, highlighting a drastic efficiency improvement with our method.

## References

[1] N. Voloch, "MSSP for 2-D sets with unknown parameters and a cryptographic application", *Contemp. Eng. Sci.*, vol. 10, no. 19, pp. 921–931, 2017.

[2] H. M. Salkin and C. A. De Kluyver, "The knapsack problem: a survey", *Naval Research Logistics Quarterly*, vol. 22, no. 1, pp. 127–144, 1975, Wiley Online Library.

[3] G. J. Woeginger and Z. Yu, "On the equal-subset-sum problem", *Information Processing Letters*, vol. 42, no. 6, pp. 299–302, 1992, doi:https://doi.org/10.1016/0020-0190(92)90226-L.

[4] A. Antonopoulos, A. Pagourtzis, S. Petsalakis, and M. Vasilakis, "Faster algorithms for k-subset sum and variations", *Journal of Combinatorial Optimization*, vol. 45, no. 1, p. 24, 2023, Springer.

[5] M. Cieliebak, S. J. Eidenbenz, A. Pagourtzis, and K. Schlude, "On the Complexity of Variations of Equal Sum Subsets", *Nord. J. Comput.*, vol. 14, no. 3, pp. 151–172, 2008.

[6] W. Wang and J. Nguyen, "The k-subset sum problem over finite fields", *Finite Fields and Their Applications*, vol. 51, pp. 204–217, 2018, Elsevier.

[7] A. Caprara, H. Kellerer, and U. Pferschy, "The multiple subset sum problem", *SIAM Journal on Optimization*, vol. 11, no. 2, pp. 308–319, 2000, SIAM.

[8] G. H. Hardy and S. Ramanujan, "Asymptotic formulaæ in combinatory analysis", *Proceedings of the London Mathematical Society*, vol. 2, no. 1, pp. 75–115, 1918, Wiley Online Library.

[9] H. Rademacher, "A convergent series for the partition function p(n)", *Proceedings of the National Academy of Sciences*, vol. 23, no. 2, pp. 78–84, 1937, National Acad Sciences.

[10] E. T. Bell, "Exponential numbers", *The American Mathematical Monthly*, vol. 41, no. 7, pp. 411–419, 1934, Taylor & Francis.

[11] É. Bertin and M. Clusel, "Generalized extreme value statistics and sum of correlated variables", *Journal of Physics A: Mathematical and General*, vol. 39, no. 24, pp. 7607–7619, 2006, doi:10.1088/0305-4470/39/24/001.

[12] B. Girard, S. Griffiths, and Y. O. Hamidoune, "k-sums in abelian groups", *Combinatorics, Probability and Computing*, vol. 21, no. 4, pp. 582–596, 2012, doi:10.1017/s0963548312000168.