Sawtooth Software

RESEARCH PAPER SERIES

The Handy (But Little Known) Q-Sort Rating Method: Plus anchored and HB-MNL extensions

Bryan Orme
Sawtooth Software, Inc.

© Copyright 2025, Sawtooth Software, Inc.
3210 N. Canyon Rd., Provo, Utah
+1 801 477 4700
www.sawtoothsoftware.com

The Handy (But Little Known) Q-Sort Rating Method: Plus anchored and HB-MNL extensions

Bryan Orme[1], Sawtooth Software

May 2025

Introduction and Motivation

Never heard of Q-sort? Don't feel bad. I hadn't heard about it either until my colleague at Sawtooth, Keith Chrzan, told me about it some dozen years ago. It originated over seventy years ago (Stephenson 1953).

In short, it's a rating scale where we don't let respondents straightline us. We force them to use a prescribed number of 5s, 4s, 3s, etc. More on that below…

Keep reading! It's SUPER easy to understand Q-sort and (no kidding) the grade-school-level math behind it. It's surprising how simple yet effective it is!

If you've read much Sawtooth literature at all, you know the standard arguments about standard rating scales being *bad* (straightlining, scale use bias) and MaxDiff being *good* (better discrimination, avoiding scale use bias). Well, think of Q-sort as much better than standard rating scales and almost as good as MaxDiff. As long as:

1. You're dealing with about 8 to 18 items.
2. You're using a survey platform that can do dynamic (constructed) lists, like Sawtooth's Discover or Lighthouse Studio platforms.

You might want to use Q-sort if you find MaxDiff too complicated to design, field, or analyze. (But, remember, it's limited to about 8 to 18 items!). More than about 18 items, and it becomes difficult to administer.

Example Q-Sort

Say we want to ask respondents to rate nine ice cream flavors. We randomize the list of flavors and first show a single-select question:

Which of these flavors of ice cream do you prefer the MOST? <select one>

Vanilla
Chocolate
Strawberry

[1] With thanks to Keith Chrzan for his guidance and suggestions regarding Q-sort.

Cookies and Cream
Mint Chocolate Chip
Chocolate Chip Cookie Dough
Butter Pecan
Rocky Road
Salted Caramel

Let's say the respondent picks **Mint Chocolate Chip** as their favorite. We assign that item a **5** on the rating scale, remove it from the list, and ask a second question with the remaining items shown:

Among the remaining flavors, which do you prefer the LEAST? (select one)

Vanilla
Chocolate
Strawberry
Cookies and Cream
Chocolate Chip Cookie Dough
Butter Pecan
Rocky Road
Salted Caramel

Let's say the respondent picks **Butter Pecan** as their least favorite. We assign that item a **1** on the rating scale, remove it from the list, and ask a third question with the remaining items shown:

Among the remaining flavors, which two do you prefer the MOST? (Select two flavors)

Vanilla
Chocolate
Strawberry
Cookies and Cream
Chocolate Chip Cookie Dough
Rocky Road
Salted Caramel

Let's say the respondent picks **Cookies and Cream** and **Rocky Road** as their two most preferred among these remaining flavors. We assign those two items a **4** on the rating scale, remove them from the list, and ask a fourth question with the remaining items shown:

Among the remaining flavors, which two do you prefer the LEAST? (Select two flavors)

Vanilla
Chocolate
Strawberry
Chocolate Chip Cookie Dough
Salted Caramel

The respondent picks **Vanilla** and **Strawberry** as their two least preferred among these remaining flavors. We assign those items a **2** on the rating scale, and we're done!

Here's the distribution of the 1-5 scores we've assigned the ice cream flavors:

Score:	Ice Cream Flavors:	Frequency:
5	Mint Chocolate Chip	1
4	Cookies and Cream, Rocky Road	2
3	Chocolate, Chocolate Chip Cookie Dough, Salted Caramel	3
2	Vanilla, Strawberry	2
1	Butter Pecan	1

Notice we've forced a quasi-normal distribution to the ratings (scores). We've only allowed one 5 and one 1, not letting respondents straightline us. Is it perfect? No. But Q-sort *works* very well in practice, as we detail further below.

Note: we illustrated the Q-sort above alternating between asking about bests and worsts. But, we could have asked the multiple stages of bests first followed by the multiple stages of worsts last. There also are ways to ask about both bests and worsts on the same screen.

How efficient is Q-sort? Well, with this 9-item list, it required just six clicks of the mouse. If we were asking the respondent to rate all items in a standard ratings grid, it would have required 9 clicks (one for each item). With MaxDiff, we would generally have recommended about four to six MaxDiff questions showing four flavors per question. Because MaxDiff asks for best and worst in each question, that's 8 to 12 clicks for MaxDiff.

So, in terms of the number of clicks per respondent: Q-sort < ratings < MaxDiff. But, not all clicks are equal in terms of time and cognitive difficulty. My colleague Keith Chrzan fielded a comparison study and found that for a 10-item study, ratings took 38 seconds, Q-sort took 75 seconds, and MaxDiff took 171 seconds (Chrzan and Golovashkina 2006). We should note that in their MaxDiff cell, respondents saw each item a full four times, whereas Sawtooth generally recommends that showing each item 3x is probably enough.

How Well Does Q-Sort Work?

Chrzan and Golovashkina compared six stated importance approaches for survey research: 1) Importance ratings, 2) Constant sum, 3) MaxDiff, 4) Q-sort, 5) Unbounded ratings, and 6) Magnitude estimation. 1284 respondents completed their survey, with each respondent randomly receiving two of the six stated importance methods (Chrzan and Golovashkina 2006). After testing the methods for between-item discrimination, predictive validity, and between-group discrimination, they found that Importance ratings and the Unbounded ratings performed the worst. MaxDiff performed the best, though it was the longest task for respondents to complete. Q-sort had the advantage of being much quicker to administer than MaxDiff, with results nearly as good as MaxDiff.

An Anchored Q-Sort Approach

One of the complaints against MaxDiff that would also apply to Q-sort is that we don't learn whether a respondent likes the items or not in any absolute sense. To alleviate that, anchored MaxDiff has become popular among Sawtooth users. After asking the MaxDiff questions, we subsequently show all (or a subset of the items) and ask respondents if each item is "important/not important," a "buy/not buy," or a "like/don't like". With that information we can scale the scores with respect to the anchor. We can give the anchor a reference score of 0 (leading to positive and negative scores), or alternatively setting the anchor to 100 (leading to all positive scores).

Anchored scaling is also useful when applying TURF (total unduplicated reach and frequency) optimization, as an item exceeding the anchor can be said to reach the respondent.

It occurs to me (I don't know if this idea is new) that we can also establish an anchor in Q-sort (after asking the Q-sort sequence) by showing the list of items[2] ordered from best to worst and asking the respondent to indicate for each whether it is "important/not important," "like/don't like", etc. Though, it's possible for the respondent to answer in a way that was inconsistent with their original Q-sort ratings, which could make it challenging to resolve the conflicts without using a more advanced method of score estimation, such as MNL as described below.

Score Estimation via HB-MNL or LC-MNL

(OK, now moving beyond grade-school math…)

In chatting with my colleague Keith as I prepared my thoughts for this article, he suggested that we could analyze Q-sort data with hierarchical Bayes MNL (multinomial logistic regression) or LC-MNL (latent class MNL). One simple approach is to explode the Q-sort data into all paired comparisons (choices between two items) for which we know the order of preference.

[2] Or, as the list of items grows relatively large, a strategically chosen subset of the list, such as 7 out of 14 items arranged from best to worst.

In the example above with 9 ice cream flavors, we can explode the data into 22 paired comparisons where we know which item is preferred. For example, the first eight paired comparisons are that the respondent prefers Mint Chocolate Chip to the other eight flavors, etc.

If we are using anchored Q-sort, we code the anchor "item" as if it were the 10th (reference) item in the design matrix. With dummy-coding, that anchor item is the reference item and is constrained to have a utility of 0. Items that are "important" get positive utility scores for the respondent and items that are "not important" get negative utility scores. We code the additional paired comparisons between the ice cream flavors and the anchor item. So, we append 9 additional paired comparison choice tasks involving the anchor item to the 22 paired comparisons from the core Q-sort stages, for a total of 31 paired comparison choice tasks per respondent.

Oh, and don't bother looking at the HB RLH fit statistic, as Q-sort coded as a paired comparison choice experiment makes it look like each respondent has been perfectly consistent. Even with augmented paired comparisons against the anchor threshold, where there's opportunity for respondents to demonstrate inconsistency, there's probably just not enough information to distinguish very well between consistent and near-random answering respondents.

Why analyze Q-sort with HB-MNL?

1) To leverage information from the distribution of respondents to strengthen the individual-level preference estimates. For example, with the standard Q-sort scoring, there are many ties among middling items. HB estimation breaks those ties by imputing information from the upper-level model (the population estimates of means and covariances among the items).
2) To allow for probabilistic statements (e.g., simulations) about the likelihood that respondents would pick one item over another, or relative shares of choice among competitive sets of items.
3) As a way to handle potential response inconsistencies if applying anchored Q-sort.

Why analyze Q-sort with LC-MNL?

1) To use the powerful and reliable latent class approach to finding segments of respondents who have similar preferences.
2) (Just like HB-MNL) to allow for probabilistic statements (e.g., simulations) about the likelihood that respondents would pick one item over another, or relative shares of choice among competitive sets of items. As a way to handle potential response inconsistencies if applying anchored Q-sort.

Appendix: Suggested Forced Distributions by Number of Items

Below, I suggest forced Q-sort distributions that never ask respondents to select more than three items at a time in any one question.

#Items	Distribution		
6	1:1:2:1:1		
7	1:1:2:2:1		
8	1:2:2:2:1		
9	1:2:3:2:1		
10	1:2:4:2:1		
11	1:2:5:2:1	or	1:2:3:2:2:1
12	1:3:4:3:1	or	1:2:3:3:2:1
13	1:3:5:3:1	or	1:2:4:3:2:1
14	1:3:6:3:1		
15	1:2:3:3:3:2:1		
16	1:2:3:4:3:2:1		
17	1:2:3:5:3:2:1		
18	1:2:3:6:3:2:1		

References

Chrzan, Keith and Natalia Golovashkina (2006), "An Empirical Test of Six Stated Importance Measures." *International Journal of Market Research,* 48:6. Available at: https://sawtoothsoftware.com/resources/technical-papers/an-empirical-test-of-six-stated-importance-measures

Stephenson, W. (1953) The Study of Behavior: The Q-Technique and Its Methodology. Chicago: University of Chicago Press.