A computational efficient way to compute P-RRM attribute vectors

1 Motivation

RRM models postulate that regret is experienced by the need to trade-off attributes across alternatives in the choice set. To compute the regret of alternative i, the performance of alternative i is compared to the performance of all other alternatives in the choice set with regard to attribute m. To compute the overall levels of regret, regrets are summed over all the attributes. As a consequence of the comparing nature of RRM models, to compute the regret levels of all alternatives in a data set is proportional to N x J x (J-1) x M, where N is the number of choice observations, J is the choice set size, and M the number of attributes. Thus, the computational efforts grow linear with N, and M, and quadratically with J. This quadratic increase in computational efforts hampers application of RRM models in large scale applications. Generally, choice set sizes in large scale applications are beyond what is computationally feasible.

Recently, a computationally more efficient RRM model has been proposed: the P-RRM model, see equation 1. Due to the piece-wise linear-attribute level regret function of this model, it is possible to compute the sum over the pairwise comparisons – called the P-RRM attribute vector x_{im}^{P-RRM} – *prior* to estimation, rather than a multitude of times during each iteration step of the maximum likelihood simulation. As such, this model is better suitable for large scale applications. The fact that the pairwise comparisons only need to be evaluated once implies a large decrease in computational efforts. For example, while it costs 3 days to estimate a Classical RRM model on a data set consisting of about 800 observations, 700 alternatives, and 10 attributes. It takes less than 10 minutes to estimate a P-RRM model on that same data.

$$
\begin{aligned}
&R_i^{P-RRM} = \sum_m \beta_m x_{im}^{P-RRM} \quad where\ x_{im}^{P-RRM} = \begin{cases} \sum_{j \neq i} \max\left(0, x_{jm} - x_{im}\right) \ if\ \beta_m > 0 \\ \sum_{j \neq i} \min\left(0, x_{jm} - x_{im}\right) \ if\ \beta_m < 0 \end{cases} \\
&P_i^{P-RRM} = \frac{e^{-R_i^{P-RRM}}}{\sum_J e^{-R_j^{P-RRM}}}
\end{aligned}
\tag{1}
$$

However, despite that the P-RRM attribute vector x_{im}^{P-RRM} only needs to be computed once to be able to estimate a P-RRM model, the computational time required to compute the P-RRM attribute vector is still quadratic with the choice set size. When dealing with very large number of alternatives, such as typically is the case with the mode-destination-time of day models in large scale travel demand models, computational time to compute the P-RRM attribute vector may exceed computational capacity boundaries. In the textbox below the effect of choice set size is illustrated.

Station choice model is of the following size: N = 800, J = 700, M = 10

$$800 \cdot 700^2 \cdot 10 \simeq 4 \cdot 10^9 \text{ pairwise comparisons} \quad \propto \quad 3.5 \text{ minutes (using 6-cores)}$$

The mode-destination model is considerably larger than the station choice model:
N = 36 000, J = 9000, M = 10

$$36\,000 \cdot 9000^2 \cdot 10 \simeq 29\,000 \cdot 10^9 \text{ pairwise comparisons} \propto \frac{29\,000}{4} \cdot 3.5 \text{ minutes} = 18 \text{days}$$

The substantial increase in computational times is dominated by the quadratic term of the choice set size, which increases computational efforts by a factor of: $\frac{9000^2}{700^2} = 165$.

2 A new, efficient way to compute the P-RRM attribute vector

Let us suppose that the choice set of decision maker n consists of J_n alternatives. Furthermore, suppose we know the signs of the taste parameters. Due to the linear nature of the P-RRM model, this implies that we know the rank-order of each alternative with regard to the performance of attribute *m*. Let *k* denote the rank of the alternatives, where alternative *k = 1* is the best performing alternative (e.g. highest level of comfort) and alternative *k = K* is the worst performing alternative (e.g. lowest level of comfort). Then, the observed regret of alternative *k* caused by attribute *m*, denoted r_{kmn}, for alternatives *k=1…K* is given in Equation 2. Note we dropped the subscript n from legibility.

$$
\begin{aligned}
r_{1m} &= 0 \\
r_{2m} &= \beta_m \left[x_{1m} - x_{2m} \right] \\
r_{3m} &= \beta_m \left[x_{1m} - x_{3m} \right] + \beta_m \left[x_{2m} - x_{3m} \right] \\
&\vdots \\
r_{km} &= \beta_m \left[x_{1m} - x_{km} \right] + \beta_m \left[x_{2m} - x_{km} \right] + \ldots + \beta_m \left[x_{k-1\,m} - x_{km} \right] \\
&\vdots \\
r_{Km} &= \beta_m \left[x_{1m} - x_{Km} \right] + \beta_m \left[x_{2m} - x_{Km} \right] + \ldots + \beta_m \left[x_{K-1\,m} - x_{Km} \right]
\end{aligned}
\tag{2}
$$

Hence, the regret of rank-ordered alternative *k* with regard to attribute *m* takes the following sequence (Equation 3).

$$
r_{km} = \beta_m \sum_{w=1}^{k-1} \left[x_{wm} - x_{km} \right] \tag{3}
$$

The key step we make is that we decompose Equation 3 into its 'principle components': $\left[x_{1m} - x_{wm} \right]$. That is, we 'remove' all 'redundant' pairwise comparisons, and replace these by principle components. To illustrate what we mean, suppose alternative k = 4 is considered, and compared to alternative k = 3, then the regret experienced by this pairwise comparison is: $r_{4m} = \beta_m \left[x_{3m} - x_{4m} \right]$. This term can however also be written in terms of its principle components as: $\beta_m \left[\left(x_{1m} - x_{4m} \right) - \left(x_{1m} - x_{3m} \right) \right]$. The regret of rank-ordered alternative *k* with regard to attribute *m* decomposed in its principle components is given in Equation 4

$$
r_{km} = \beta_m \left\{ \left(k - 1 \right) \left[x_{1m} - x_{km} \right] - \sum_{w=2}^{k-1} \left[x_{1m} - x_{wm} \right] \right\} \tag{4}
$$

Equation 5 shows equation 4 in matrix form. It shows that we can express all regrets in terms of the principle components. This implies that we do not have to compute the large number of basically redundant pairwise comparisons. In fact, we only need to compute the attribute level differences as compare to the best performing alternative, and the rank order structure. In this form the computational efforts required to compute the P-RRM matrix are linear with choice set size; rather than quadratic. This has major implications for the required computational time to compute the P-RRM attribute vector in the context of very large choice sets.

$$\begin{bmatrix} r_{2m} \\ r_{3m} \\ \vdots \\ r_{km} \\ \vdots \\ r_{Km} \end{bmatrix} = \beta_m \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & 0 & 0 & 0 & 0 \\ \vdots & & \ddots & & & \\ -1 & -1 & -1 & k-1 & 0 & 0 \\ \vdots & & & & \ddots & \\ -1 & -1 & -1 & -1 & -1 & K-1 \end{bmatrix} \begin{bmatrix} \left[x_{1m} - x_{2m}\right] \\ \left[x_{1m} - x_{3m}\right] \\ \vdots \\ \left[x_{1m} - x_{km}\right] \\ \vdots \\ \left[x_{1m} - x_{Km}\right] \end{bmatrix} \tag{5}$$

So far we have assumed complete rank-order across alternatives exists. However, in practice this is often not the case due to the presence of alternatives having the same attribute levels. Therefore, we need to generalize equation 5, to cope with this situation. Let C_k denote the set of alternatives having rank *k*, and let Z_k denote the cardinality of C_k. Then, the regret associated with all alternatives with rank *k*, denoted $\underset{j \in C_k}{r_m}$, is given in equation 6. Note that the presence of alternatives having the same attribute levels even further improves computational speed of this method.

$$\begin{bmatrix} \underset{j\in C_2}{r_m} \\ \underset{j\in C_3}{r_m} \\ \vdots \\ \underset{j\in C_k}{r_m} \\ \vdots \\ \underset{j\in C_K}{r_m} \end{bmatrix} = \begin{bmatrix} Z_1 & 0 & 0 & 0 & 0 & 0 \\ -Z_1 & \sum_{k=1}^{2} Z_k & 0 & 0 & 0 & 0 \\ \vdots & & \ddots & & & \\ -Z_1 & -Z_2 & \cdots & \sum_{k=1}^{k-1} Z_k & 0 & 0 \\ \vdots & & & & \ddots & \\ -Z_1 & -Z_2 & \cdots & -Z_k & \cdots & \sum_{k=1}^{K-1} Z_k \end{bmatrix} \begin{bmatrix} [x_{1m} - x_{2m}] \\ [x_{1m} - x_{3m}] \\ \vdots \\ [x_{1m} - x_{km}] \\ \vdots \\ [x_{1m} - x_{Km}] \end{bmatrix} \tag{6}$$

3 Results

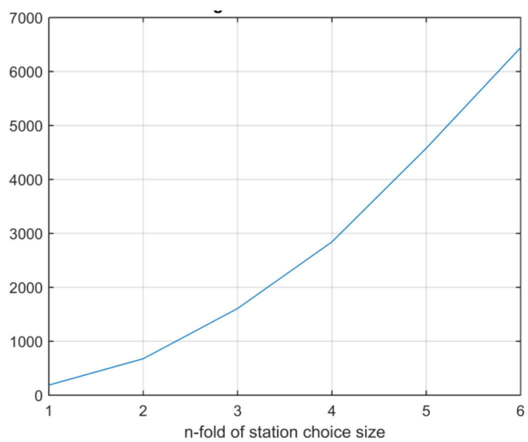
Our test set up to test the improvement in computational speed is as follows. We replicate alternatives of the station choice data up to 10 times.

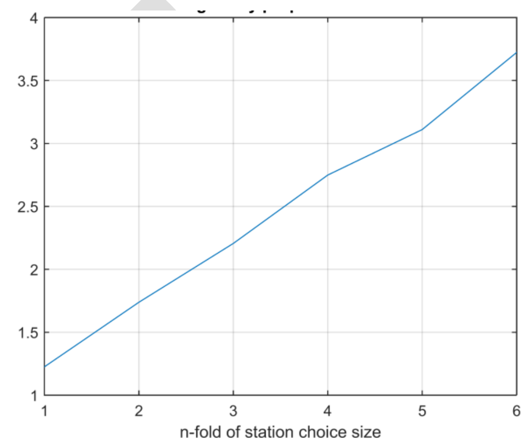
Table 1 shows the obtained computational times:

Data set size			Time to compute the P-RRM matrix	
N	*J*	*M*	pairwise-method [sec]	new method [sec]
791	1 x 684	10	218	1.2
791	2 x 684	10	875	1.9
791	3 x 684	10	1922	2.3
791	4 x 684	10	~3500	2.8
791	5 x 684	10	~5450	3.0
791	6 x 684	10	~7850	3.7
791	7 x 684	10	~10700	5.0
791	8 x 684	10	~14 000	5.6

791	9 x 684	10	~17 700	6.4
791	10 x 684	10	~21 800	7.0
36 000	9000	10	~1.7 10^6 ~3 weeks	~415 ~7 minutes

Table 1





As can be seen, the time to compute the P-RRM attribute vector reduces drastically by using the newly proposed method. It takes minutes rather than weeks to compute the P-RRM-matrix in the context of large choice sets. However, partly this huge improvement is due to that replication of the station choice data did not result in new attribute level. Thus, it basically doubled all Z_k's. Nonetheless, computation of similar sizes of randomly generated data set takes is still very fast, see Table 2.

Data set size			
N	*J*	*M*	new method [sec]
791	1 x 684	10	18
791	2 x 684	10	20
791	3 x 684	10	22
791	4 x 684	10	24
791	5 x 684	10	27

36 000	9000	10	~3200 ~53 minutes

Table 2