A Benders decomposition for maximum simulated likelihood estimation of advanced discrete choice models

Tom Haering
Claudia Bongiovanni
Michel Bierlaire

STRC conference paper 2022

May 23, 2022

STRC 22nd Swiss Transport Research Conference
Monte Verità / Ascona, May 18-20, 2022
A Benders decomposition for maximum simulated likelihood estimation of advanced discrete choice models

Tom Haering
TRANSP-OR
ETH Lausanne
CH-1003 Lausanne
tom.haering@epfl.ch

Claudia Bongiovanni
TRANSP-OR
ETH Lausanne
CH-1003 Lausanne
claudia.bongiovanni@epfl.ch

Michel Bierlaire
TRANSP-OR
ETH Lausanne
CH-1003 Lausanne
michel.bierlaire@epfl.ch

May 23, 2022

Abstract

In this paper, we formulate a mixed integer linear program (MILP) for the simulated maximum likelihood estimation (MLSE) problem and devise a Benders decomposition approach to speed up the solution process. This framework can be applied to any advanced discrete choice model and exploits total unimodularity to keep the master problem linear in the decomposition. The proposed decomposition approach is benchmarked against the original MILP formulation and PandasBiogeme. Computational experiments are performed on a binary logit mode choice model with up to 200 respondents. Results show that the Benders decomposition approach solves instances on average 35 and up to 100 times faster than the MILP while maintaining high quality solutions. We furthermore give detailed descriptions of ideas for future enhancements of the approach.

Keywords
benders decomposition, discrete choice, maximum likelihood estimation, mixed integer linear programming, simulation
A Benders decomposition for maximum simulated likelihood estimation of advanced DCMs

May 23, 2022

Contents

List of Tables .................................................. 1
List of Figures .................................................. 2

1 Background .................................................. 3

2 Methodology .................................................. 5
  2.1 MILP formulation ........................................ 5
  2.2 Benders decomposition approach ....................... 6

3 Results and discussion ..................................... 9
  3.1 Application to a mode choice problem ............... 9
  3.2 Application to a continuous pricing problem ...... 11
  3.3 Large numbers of draws ............................... 13
  3.4 Decomposition as a warm start ....................... 14

4 Plan for future research .................................... 16
  4.1 Modeling framework ................................... 16
  4.2 Decomposition methods ................................ 17
  4.3 Linearization methods ................................ 18

5 Conclusion .................................................. 21

6 References .................................................. 21

List of Tables

1 Comparing our decomposition method with the full MILP in terms of $sLL$ and runtime
  (N = population size, R = number of draws, sLL = simulated Log-Likelihood, M = MILP, D = decomposition, T = time in sec.) ....................... 10

2 Comparing our decomposition method with the full MILP and PandasBiogeme in
terms of $LL$ (N = population size, R = number of draws, LL = Log-Likelihood, Biog = PandasBiogeme, M = MILP, D = decomposition). ............... 10

3 Comparing our decomposition method with the full MILP and PandasBiogeme in
terms of $\hat{\beta}$ (N = population size, R = number of draws, Biog = PandasBiogeme, M = MILP, D = decomposition) .......................... 11
Comparing our decomposition method with the full MILP in terms of objective, price and runtime (N = population size, R = number of draws, obj = objective, P = price, M = MILP, NL = Nonlinear, D = decomposition, R = decomp along scen, T = time in sec.)

Comparing our decomposition method with the full MILP in terms of \( sLL \) and runtime for large numbers of draws (N = population size, R = number of draws, \( sLL \) = simulated Log-Likelihood, M = MILP, D = decomposition, T = time in sec.)

Comparing our decomposition method as a warm start with the full MILP in terms of \( sLL \) and runtime (N = population size, R = number of draws, \( sLL \) = simulated Log-Likelihood, M = MILP, D = decomposition, M+ = MILP with warm start, T = time in sec.)

Compatibility between decomposition methods and linearization methods

List of Figures

1. MSLE as an MILP
2. Flowchart for Benders’ decomposition
3. MSLE - Primal subproblem
4. MSLE - Dual subproblem
5. MSLE - Master problem
6. Flowchart for column generation

1. MSLE as an MILP
2. Flowchart for Benders’ decomposition
3. MSLE - Primal subproblem
4. MSLE - Dual subproblem
5. MSLE - Master problem
6. Flowchart for column generation
1 Background

Maximum likelihood estimation (MLE) is a broadly used method to estimate the parameters, given observed data (Myung, 2003). It finds its use in many areas of mathematical statistics (Sur and Candès, 2019), physics (Hauschild and Jentschel, 2001), machine learning (Goodfellow et al., 2016) and discrete choice modeling (Ben-Akiva and Bierlaire, 2003). The latter specifically relies on the use of MLE to estimate the optimal parameters of convex and non-convex discrete choice models (Bierlaire, 1998). The state-of-the-art approach for the estimation process is to use classical non linear optimization algorithms, that require that the likelihood function to be continuous in the unknown parameters, and to have a closed form, which is also concave. These requirements are actually verified only for the estimation of logit models with linear-in-parameters utility functions. More complex models (the nested logit model, the cross-nested logit model, choice models with latent classes) are associated with non concave likelihood functions. Non linear optimization algorithms may easily be trapped in a local optimum. Models based on mixtures (including choice models with latent variables) are associated with a likelihood function that has no closed form, and involves complex integrals. Monte-Carlo simulation is therefore required to approximate it (Train, 2009). Discrete specification decisions, such as including an explanatory variable or not, or including an alternative in a nest) are handled "by hand", in the sense that all possible specifications are explicitly enumerated, and each of them estimated separately. A general implementation approach for MSLE, that is able to handle all of these cases, has been proposed in Fernández Antolín (2018), where the problem is formulated as a mixed integer linear program (MILP). The approach relaxes any assumption on the specific shape of the error term distribution and instead only assumes that it is possible to take draws. This allows it to be flexibly applied to any model. The approach is especially well suited to handle integral estimation parameters, like class membership variables in a latent class model or to implement an assisted / automatic specification approach (Fernandez-Antolin et al., 2018). With a sufficiently large number of draws, the MILP formulation guarantees convergence to global optimal solutions. However, since the complexity of MILP scales exponentially with the number of draws, the approach can currently only be applied to solving small-scale instances, i.e., with few individuals and alternatives (Pacheco et al., 2021).

Recent research studies from the literature have shown that the computational complexity resulting from simulation-based optimization approaches can be overcome by mathematical decomposition techniques (e.g. Bront et al., 2009; Koch et al., 2017). Mathematical decomposition is an optimization field that aims at exploiting specific structural properties of decision-making problems to speed-up the solution process by parallelization, size reduction, and simplification (Conejo et al., 2006). Discrete decision-making problems can be characterized broadly by the two following structural properties: (i) a set of complicating decision variables, (ii) a set of complicating constraints. Both of these properties have been addressed through specific mathematical
decomposition techniques. Namely, (i) Benders’ decomposition [Rahmaniani et al., 2020], (ii) column generation [Desaulniers et al., 2006], and (iii) Langrangian decomposition [Fisher, 1981]. For example, Benders’ decomposition has been widely applied to relax complicating variables in the integrated airline scheduling problem (e.g. Cordeau et al. 2001; Papadakos 2009). For the same problem, Yan et al. (2020) proposed a new choice-based reformulation that employs mathematical decomposition to reduce the size of the problem. Bortolomiol et al. (2021) implemented a Branch-and-Benders-Cut procedure for tackling a discrete assortment pricing problem. Similarly to Benders, Lagrangian decomposition has been widely applied to relax complicating constraints in network design problems (e.g. Heidari-Fathian and Pasandideh 2018; Gendron 2019; Shan et al. 2020) or facility location problems (e.g. Alenezy 2020; Yu et al. 2017). A recent choice-based network design problem employing Lagrangian relaxation to determine hub locations was proposed in Tiwari et al. (2021). Pacheco et al. (2018) employ a Lagrangian decomposition for choice-based optimization problems, by first duplicating the complicating variables for each simulation scenario and then adding constraints that force the duplicates to take equal values over all scenarios. In order to strengthen the decomposition and prevent loss of structure, scenarios were clustered into groups. However, the approach yielded only limited success, leading to the conclusion that this decomposition technique might not be the most appropriate for this problem. Finally, column generation has been widely applied to solve large-scale scheduling and problems (e.g. Desaulniers et al. 2002; Feillet 2010; Boyer et al. 2014). A recent choice-based multi-project scheduling and staffing problem employing column generation was proposed in Van Den Eeckhout et al. (2021).

In this work, we formulate a Benders’ decomposition approach, which speeds-up the MILP solution process for the MSLE and enables to scale-up the tackled instances. Our designed Benders’ decomposition approach exploits total unimodularity to keep the master problem linear, thus eliminating the bottleneck in computational time usually associated with Benders decomposition. The proposed approach is benchmarked against the full MILP and PandasBiogeme [Bierlaire 2020]. Although the method is general, the developed approach is currently being investigated on a binary logit mode choice model with up to 200 respondents. Initial results show that, while being very competitive in terms of computational time, our Benders’ decomposition approach produces small optimality gaps. We present the approach in section 2 and show results, together with investigations of different remedies for the optimality gaps in Section 3.
2 Methodology

In this section, we formally introduce an MILP formulation for the MSLE problem, based on the work in Fernández Antolín (2018), and a problem-specific Benders decomposition approach. Without loss of generality, the formulation is presented in the context of a multinomial logit formulation, with examples on how to extend it to other model classes, such as probit and latent class models.

2.1 MILP formulation

Consider a set of \( n = \{1, \ldots, N\} \) individuals choosing exactly one alternative among a set of \( i = \{1, \ldots, I\} \) alternatives. The data underlying the problem consists of such choices, depicted by a binary input parameter \( y_{in} \). For each alternative \( i \) we define a utility function \( U_{in} = U_{in}(\beta, \varepsilon_{in}) \), where \( \beta \) is a vector of unknown parameters, and \( \varepsilon_{in} \) is a random vector. We assume the utility to be linear in \( \beta \) and the distribution of \( \varepsilon \) such that we can generate draws from it. These error term draws are necessary to get a linear specification, and we denote them as \( \varepsilon_{inr} \), for \( R \) scenarios \( r \in \{1, \ldots, R\} \). This allows us to write the utilities \( U_{inr} \) in a deterministic way, meaning that they only depend on \( \beta \). For this exposition, we write \( U_{inr} = \sum_{k} \beta_{k} x_{ink} + \varepsilon_{inr} \), but more complex specifications are possible, if linearity in \( \beta \) is maintained. Denote by \( \omega_{inr} \) the binary decision variable that indicates whether individual \( n \) chooses alternative \( i \) in scenario \( r \). Random utility theory dictates that, in each scenario \( r \), each individual \( n \) selects the alternative \( i \) corresponding to the maximal utility \( U_{inr} \), i.e. \( \omega_{inr} = 1 \iff U_{inr} = \max_{j} U_{ijnr} \). The objective is to maximize the likelihood function, given by \( \prod_{n} \prod_{i} P_{n}(i)^{y_{in}} \), where \( P_{n}(i) \) represents the probability of individual \( n \) choosing alternative \( i \). The choice probabilities are approximated by \( P_{n}(i) \approx \frac{1}{R} \sum_{r} \omega_{inr} \) and are guaranteed to converge to the real probabilities with a...
sufficiently large number of scenarios $R$ [Pacheco et al. 2021]. Taking the log of the likelihood and replacing $P_n(i)$ by its estimator yields an objective that still contains the nonlinear term $\ln(\sum_r \omega_{inr})$. This issue is tackled by introducing the auxiliary decision variable $s_{in} = \sum_r \omega_{inr}$, which is defined in constraints ($\theta_{in}$). Similarly, an auxiliary variable $z_{in}$ is introduced to represent the piece-wise linearization of the logarithm. The latter is defined in Constraints ($\xi_{inr}$), where $L_r = (1 + r) \ln(r) - r \ln(1 + r)$ and $K_r = \ln(r) - \ln(1 + r)$ are constants representing intercepts and slopes used for the linearization. With such pre-processing steps and by ignoring the constant term $-N \ln(R)$, the objective of the problem can be rewritten as stated in Formulation [1]. The rest of the constraints model individual choices. Constraints ($\mu_{inr}$) guarantee that only one alternative can be chosen per individual and scenario. Constraints ($\kappa_{inr}$) model the utility of each alternative $i$ for individual $n$ in scenario $r$, i.e. $U_{inr}$. Constraints ($\zeta_{nr}$), which can be easily linearized using a standard big-M approach, and constraints ($\alpha_{inr}$) ensure that the choice being made corresponds to the one with the highest utility. Note that Formulation [1] is characterized by the complicating binary decision variables $\omega$.

The MILP formulation can be easily adapted to other model specifications. For example, in order to tackle a probit model, it is sufficient to add the Cholesky factor of the covariance matrix [Dow and Endersby 2004] in the right-hand side of constraints ($\kappa_{inr}$). Note that this transformation increases the number of parameters to be estimated by $I(I+1)/2$. Similarly, in order to tackle a latent class model, it is sufficient to add a class membership indicator $\gamma_{cn}$ where $c$ is the class index. For each class $c$, the constraints corresponding to making the best choice $\omega_{cinr}$ for that class are duplicated, and finally a global choice variable is defined as $\omega_{inr} = \sum_c \omega_{cinr} \gamma_{cn}$.

### 2.2 Benders decomposition approach

Combinatorial optimization problems that are characterized by complicating variables are typically tackled by a Benders decomposition approach [Benders 1962]. The logic of this approach lies in temporarily fixing the (usually integral) complicating variables to give rise to much simpler linear sub-problems to be solved. With this premise, a mathematical program is therefore split into two problems: (i) a problem containing all integer decision variables and constraints (i.e. the restricted master problem), and (ii) a problem containing all continuous decision variables and constraints (i.e. the sub-problem). The master problem is solved, and an integer-feasible solution is found, giving a lower bound on the objective value. This integer solution is successively used to solve the dualized linear sub-problem and produce: (i) feasibility cuts, if the subproblem is infeasible with the integer solution from the master problem; or (ii) optimality cuts, if the subproblem provides an optimal solution given the fixed integer solution from the master problem. The produced cuts are fed to the restricted master problem to tighten
the upper bound, after which the restricted master problem is re-solved to provide a new integer solution. The whole Benders’ process repeats up until a satisfactory solution has been found (i.e. the gap between upper and lower bound is small enough). A flowchart describing the steps for Benders’ decomposition is shown in Figure 2. For a review on Benders’ decomposition, we refer to Rahmaniani et al. (2017).

In most applications, the complicating variables are integral, resulting in the need to solve an integral master problem at each iteration. This makes Benders notorious for its slow convergence (Rahmaniani et al., 2020). In our case, we can use an elegant trick to avoid this issue: by identifying the continuous estimation parameters $\beta$ as the complicating variables and fixing them in the subproblem, the utilities of all the alternatives become fixed as well. Thus the problem of choosing the highest utility alternative simplifies to a knapsack problem, which is totally unimodular. This mathematical property allows us to drop the integrality constraints on the choice variables. Formulation 3 and Formulation 4 give the respective definitions of the primal and dual of the subproblem, while Formulation 5 describes the master problem.

Figure 2: Flowchart for Benders’ decomposition
As the linearization of Constraint \((\zeta_{nr})\) using a big-M approach no longer works when integrality constraints are relaxed, the formulation in the primal subproblem needs to be modified: The product \(\eta_{inrk} = \omega_{inr}\beta_k\) is modeled directly using Constraints \((\pi_{inr})\), \((\lambda_{inrk})\) and \((\phi_{nrk})\). This formulation is equivalent to Formulation 1. It is important to mention that, in order to guarantee total unimodularity of the primal, information about \(\beta_{\text{fixed}}\) had to be kept in its coefficient matrix, which implies it also being contained in the matrix of the dual, i.e. Constraints \((\chi_{inr})\). This means the feasible region of the dual subproblem is not constant over iterations, which might distort the Bender cuts (Rahmaniani et al., 2020). Lastly, both the primal and the dual models are fully decomposable on the individuals \(n\), as individuals select alternatives independently from each other.

Finally, the master problem reduces to finding optimal values for the estimation parameters \(\beta\). For each \(\beta_{\text{fixed}}\), after solving the dual subproblem, a Benders cut of the same type as Constraint (1) is added. The parameters of the Benders cuts are determined by the achieved objective \(L^*\) and \(\phi_{nk}^* = \sum_{ir} \lambda_{inrk}^*\). Each optimal objective value of the master problem serves as a new lower
Figure 5: MSLE - Master problem

\[
\begin{align*}
\min_{L, \beta} \quad & L \\
\text{s.t.} \quad & L \geq L^* + \sum_n \sum_k \phi_{nk}^* (\beta_k - \beta_k^{\text{fixed}}) \\
& L \geq L^{\text{best}} \\
& L \in \mathbb{R} \\
& \beta \in \mathbb{R}^K
\end{align*}
\]

bound on the objective, enforced in Constraint (2).

3 Results and discussion

3.1 Application to a mode choice problem

Our approach is tested on a binary logit model. A mode choice problem between two alternatives, public transport (pt) and car, is considered. The utilities of the alternatives consist of a systematic part \(V\) and a stochastic part \(\varepsilon\). The utilities of the alternatives for individual \(n\) are:

\[
\begin{align*}
U_{\text{car}, n} &= \beta_{\text{time}} \cdot \text{traveltime}_{\text{car}, n} + \varepsilon_{\text{car}, n} \\
U_{\text{pt}, n} &= \beta_{\text{time}} \cdot \text{traveltime}_{\text{pt}, n} + \varepsilon_{\text{pt}, n}
\end{align*}
\]

where as the stochastic part is defined by independent and identically distributed random error terms following a standard extreme value distribution, i.e. \(\varepsilon \sim \text{i.i.d. EV}(0, 1)\). The dataset is extracted from revealed preference data on mode choice collected in 1987 for the Netherlands Railways, consisting of 228 respondents (CASE, 2017). Experiments are performed using GUROBI 9.5.0 (Gurobi Optimization, LLC, 2021) on a 2.6 GHz 6-Core Intel Core i7 processor with 16 GB of RAM, with a three hour time limit per instance. Our proposed Benders approach is benchmarked against PandasBiogeme (Bierlaire, 2020) and the full MILP, in terms of objective values and runtimes. Biogeme’s objective function is the Log-Likelihood \(\text{LL} = \ln(\prod_i P_i(i)^{m_i})\), which is approximated by the simulated Log-Likelihood \(\text{sLL}\), the MILP objective. For the purpose of comparison, the \(\text{LL}\) is also evaluated for the decomposition
and the MILP using the estimated parameters. We take random subsets of individuals from the population to get instances that are manageable for the MILP.

Table 1 shows the comparison between the decomposition and the full MILP in terms of $sLL$ and computation times, while Table 3 shows the results in terms of $LL$. We highlight the following: (i) the decomposition solves the problem on average 35 and up to 100 times faster, (ii) comparing the optimal solution values for the full MILP and our decomposition reveals gaps in optimality, which are small for the objective value but larger for the estimated parameters, and (iii) increasing the number of draws reduces the optimality gap between the exact solution (PandasBiogeme) and the approximation (MILP and decomposition).

Table 1: Comparing our decomposition method with the full MILP in terms of $sLL$ and runtime ($N = \text{population size}, R = \text{number of draws}, sLL = \text{simulated Log-Likelihood}, M = \text{MILP}, D = \text{decomposition}, T = \text{time in sec.}$)

<table>
<thead>
<tr>
<th>N</th>
<th>R</th>
<th>sLL-M</th>
<th>sLL-D</th>
<th>Gap [%]</th>
<th>T-M</th>
<th>T-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>50</td>
<td>-12.607</td>
<td>-12.658</td>
<td>-0.40</td>
<td>64.942</td>
<td>10.061</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
<td>-12.212</td>
<td>-12.485</td>
<td>-0.38</td>
<td>403.694</td>
<td>9.902</td>
</tr>
<tr>
<td>50</td>
<td>50</td>
<td>-30.848</td>
<td>-31.030</td>
<td>0.59</td>
<td>286.679</td>
<td>29.780</td>
</tr>
<tr>
<td>50</td>
<td>100</td>
<td>-30.461</td>
<td>-31.040</td>
<td>1.90</td>
<td>1558.604</td>
<td>65.006</td>
</tr>
<tr>
<td>50</td>
<td>200</td>
<td>-30.566</td>
<td>-30.692</td>
<td>0.41</td>
<td>5375.655</td>
<td>98.206</td>
</tr>
<tr>
<td>100</td>
<td>50</td>
<td>-65.204</td>
<td>-65.801</td>
<td>0.92</td>
<td>2820.229</td>
<td>28.781</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>-65.784</td>
<td>-67.419</td>
<td>2.49</td>
<td>4346.067</td>
<td>274.163</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>-65.784</td>
<td>-66.018</td>
<td>0.49</td>
<td>10800+</td>
<td>295.741</td>
</tr>
</tbody>
</table>

Table 2: Comparing our decomposition method with the full MILP and PandasBiogeme in terms of $LL$ ($N = \text{population size}, R = \text{number of draws}, LL = \text{Log-Likelihood}, Biog = \text{PandasBiogeme}, M = \text{MILP}, D = \text{decomposition}$)

<table>
<thead>
<tr>
<th>N</th>
<th>R</th>
<th>LL-Biog</th>
<th>LL-M</th>
<th>Gap [%]</th>
<th>LL-D</th>
<th>Gap [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>50</td>
<td>-12.303</td>
<td>-12.444</td>
<td>-1.15</td>
<td>-12.493</td>
<td>-1.55</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
<td>-12.303</td>
<td>-12.395</td>
<td>0.75</td>
<td>-12.411</td>
<td>-0.88</td>
</tr>
<tr>
<td>20</td>
<td>200</td>
<td>-12.303</td>
<td>-12.378</td>
<td>0.61</td>
<td>-12.463</td>
<td>-1.30</td>
</tr>
<tr>
<td>50</td>
<td>50</td>
<td>-30.265</td>
<td>-30.326</td>
<td>0.20</td>
<td>-30.683</td>
<td>-1.38</td>
</tr>
<tr>
<td>50</td>
<td>100</td>
<td>-30.265</td>
<td>-30.326</td>
<td>0.20</td>
<td>-30.481</td>
<td>-0.72</td>
</tr>
<tr>
<td>50</td>
<td>200</td>
<td>-30.265</td>
<td>-30.325</td>
<td>0.20</td>
<td>-30.283</td>
<td>-0.06</td>
</tr>
<tr>
<td>100</td>
<td>50</td>
<td>-64.883</td>
<td>-64.898</td>
<td>0.02</td>
<td>-65.396</td>
<td>-0.79</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>-64.883</td>
<td>-64.883</td>
<td>0.00</td>
<td>-66.031</td>
<td>-1.77</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>-64.883</td>
<td>-64.893</td>
<td>0.02</td>
<td>-64.925</td>
<td>-0.06</td>
</tr>
<tr>
<td>200</td>
<td>50</td>
<td>-122.689</td>
<td>-122.735</td>
<td>-0.04</td>
<td>-122.690</td>
<td>0.00</td>
</tr>
<tr>
<td>200</td>
<td>100</td>
<td>-122.689</td>
<td>-122.920</td>
<td>0.19</td>
<td>-122.739</td>
<td>-0.04</td>
</tr>
<tr>
<td>200</td>
<td>200</td>
<td>-122.689</td>
<td>-123.342</td>
<td>0.53</td>
<td>-122.721</td>
<td>0.03</td>
</tr>
</tbody>
</table>
Table 3: Comparing our decomposition method with the full MILP and PandasBiogeme in terms of $\beta$ ($N =$ population size, $R =$ number of draws, Biog = PandasBiogeme, $M =$ MILP, $D =$ decomposition)

<table>
<thead>
<tr>
<th>$N$</th>
<th>$R$</th>
<th>Beta-Biog</th>
<th>Beta-MILP</th>
<th>Gap [%]</th>
<th>Beta-D</th>
<th>Gap [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>50</td>
<td>-1.558</td>
<td>-1.048</td>
<td>32.72</td>
<td>-0.97</td>
<td>37.71</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
<td>-1.558</td>
<td>-1.143</td>
<td>26.62</td>
<td>-1.11</td>
<td>28.77</td>
</tr>
<tr>
<td>20</td>
<td>200</td>
<td>-1.558</td>
<td>-1.182</td>
<td>24.11</td>
<td>-2.16</td>
<td>38.67</td>
</tr>
<tr>
<td>50</td>
<td>50</td>
<td>-1.41</td>
<td>-1.223</td>
<td>13.26</td>
<td>-0.935</td>
<td>33.64</td>
</tr>
<tr>
<td>50</td>
<td>100</td>
<td>-1.41</td>
<td>-1.223</td>
<td>13.26</td>
<td>-1.783</td>
<td>-26.46</td>
</tr>
<tr>
<td>50</td>
<td>200</td>
<td>-1.41</td>
<td>-1.223</td>
<td>13.22</td>
<td>-1.307</td>
<td>7.26</td>
</tr>
<tr>
<td>100</td>
<td>50</td>
<td>-0.948</td>
<td>-0.889</td>
<td>6.23</td>
<td>-0.612</td>
<td>35.48</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>-0.948</td>
<td>-0.943</td>
<td>0.53</td>
<td>-0.451</td>
<td>52.4</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>-0.948</td>
<td>-0.899</td>
<td>5.21</td>
<td>-0.85</td>
<td>10.33</td>
</tr>
<tr>
<td>200</td>
<td>50</td>
<td>-1.31</td>
<td>-1.39</td>
<td>-6.11</td>
<td>-1.322</td>
<td>-0.92</td>
</tr>
<tr>
<td>200</td>
<td>100</td>
<td>-1.31</td>
<td>-1.49</td>
<td>-13.73</td>
<td>-1.393</td>
<td>-6.33</td>
</tr>
<tr>
<td>200</td>
<td>200</td>
<td>-1.31</td>
<td>-1.021</td>
<td>22.04</td>
<td>-1.377</td>
<td>-5.11</td>
</tr>
</tbody>
</table>

Although Benders decomposition is an exact approach, our formulation contains mathematical aspects that may currently prevent the convergence to the real global optimum. As mentioned in the methodology, a possible explanation for the deviations is the fact that information about the master variables is maintained in the coefficient matrix of the dual. Other explanations include numerical issues, stemming for example from the linearization of the logarithm or the way certain solvers handle specific constraints.

### 3.2 Application to a continuous pricing problem

The suspicion that the log-linearization might influence the Benders cuts leads us to an immediate first attempt to fix it: apply the framework to a choice based optimization problem that does not rely on piece-wise linearization of the objective. A very important problem that displays this
characteristic is the continuous pricing problem (CPP). It can be written as follows:

$$\max_{p, \omega, U, H} \sum_{n} \sum_{r} \sum_{i} \frac{1}{R} \theta_{in} p_{i} \omega_{inr}$$

s.t.

$$\sum_{i} \omega_{inr} = 1 \quad (\mu_{inr})$$

$$H_{nr} = \sum_{i} U_{inr} \omega_{inr} \quad (\zeta_{inr})$$

$$H_{nr} \geq U_{inr} \quad (\alpha_{inr})$$

$$U_{inr} = \sum_{k \neq l} \beta_{k} \chi_{ink} + \beta_{l} p_{i} + \epsilon_{inr} \quad (\kappa_{inr})$$

$$\omega \in \{0, 1\}$$

$$p, U, H \in \mathbb{R}$$

applying the exact same methodology as for the MSLE problem, we arrive at the following primal subproblem:

$$\max_{p, \omega, U, H} \sum_{n} \sum_{r} \sum_{i} \frac{1}{R} \theta_{in} p_{i} \chi_{inr}$$

s.t.

$$\sum_{i} \omega_{inr} = 1 \quad (\mu_{inr})$$

$$\sum_{k} \beta_{k} \chi_{ink} - H_{nr} \leq -\epsilon_{inr} \quad (\alpha_{inr})$$

$$H_{nr} - \sum_{ik} \eta_{inr} \chi_{ink} \leq \sum_{i} \omega_{inr} \epsilon_{inr} \quad (\zeta_{inr})$$

$$\eta_{inr} = p_{i}^{fixed} \omega_{inr} \quad (\lambda_{inr})$$

$$\chi_{inr} + \omega_{inr} = 1 \quad (\pi_{inr})$$

$$\eta_{inr} + p_{i}^{fixed} \chi_{inr} = p_{i}^{fixed} \quad (\lambda_{inr})$$

$$p_{i} - \sum_{i} \eta_{inr} = 0 \quad (\varphi_{inr})$$

$$\omega, \chi \in \mathbb{R}_{\geq 0}$$

$$p, \eta, H \in \mathbb{R}$$

We can apply the same Netherlands mode choice data as in the MSLE problem to the CPP, with the slight change that the utilities are now defined as follows:

$$U_{car, n} = \beta_{time} \cdot \text{traveltime}_{\text{car}, n} + \beta_{cost} \cdot \text{cost}_{\text{car}, n} + \epsilon_{\text{car}, n}$$

$$U_{pt, n} = \text{ASC}_{\text{pt}} + \beta_{time} \cdot \text{time}_{\text{pt}, n} + \beta_{cost} \cdot \text{cost}_{\text{pt}, n} + \epsilon_{\text{pt}, n}$$
We now first estimate the optimal $\beta$-parameters using Biogeme and then replace the attribute cost$_{pt}$ by a variable $p_{pt}$ which is to be optimized. The problem can be thus seen as the rail company deciding on the optimal price for their service. Running experiments with the above framework leads to the results shown in Table 4. Clearly, the gaps are still existent, proving that the linearization of the log in the MSLE problem is not the (only) source of the issue.

Table 4: Comparing our decomposition method with the full MILP in terms of objective, price and runtime ($N =$ population size, $R =$ number of draws, obj = objective, $P =$ price, $M =$ MILP, NL = Nonlinear, D = decomposition, $R =$ decomp along scen, $T =$ time in sec.)

<table>
<thead>
<tr>
<th>N</th>
<th>R</th>
<th>obj-MILP</th>
<th>obj-D</th>
<th>Gap [%]</th>
<th>P-MILP</th>
<th>P-D</th>
<th>Gap [%]</th>
<th>$T$-MILP</th>
<th>$T$-MILP (NL)</th>
<th>$T$-D</th>
<th>$T$-D (R)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>100</td>
<td>202.642</td>
<td>201.712</td>
<td>0.46</td>
<td>28.302</td>
<td>26.576</td>
<td>6.1</td>
<td>37</td>
<td>7</td>
<td>21</td>
<td>65</td>
</tr>
<tr>
<td>20</td>
<td>200</td>
<td>200.901</td>
<td>200.185</td>
<td>0.36</td>
<td>30.03</td>
<td>28.721</td>
<td>4.36</td>
<td>205</td>
<td>252</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>50</td>
<td>50</td>
<td>440.686</td>
<td>437.243</td>
<td>0.78</td>
<td>28.579</td>
<td>29.989</td>
<td>-4.94</td>
<td>55</td>
<td>24</td>
<td>27</td>
<td>20</td>
</tr>
<tr>
<td>50</td>
<td>100</td>
<td>431.088</td>
<td>426.669</td>
<td>1.03</td>
<td>28.99</td>
<td>27.778</td>
<td>4.18</td>
<td>241</td>
<td>235</td>
<td>62</td>
<td>51</td>
</tr>
<tr>
<td>50</td>
<td>200</td>
<td>429.605</td>
<td>429.108</td>
<td>0.12</td>
<td>28.574</td>
<td>28.655</td>
<td>-0.28</td>
<td>1022</td>
<td>2125</td>
<td>163</td>
<td>323</td>
</tr>
<tr>
<td>100</td>
<td>50</td>
<td>990.026</td>
<td>988.732</td>
<td>0.13</td>
<td>29.118</td>
<td>28.944</td>
<td>0.6</td>
<td>252</td>
<td>202</td>
<td>31</td>
<td>147</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>977.606</td>
<td>976.149</td>
<td>0.15</td>
<td>30.099</td>
<td>29.925</td>
<td>0.58</td>
<td>1224</td>
<td>595</td>
<td>69</td>
<td>315</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>978.589</td>
<td>976.932</td>
<td>0.17</td>
<td>30.106</td>
<td>30.185</td>
<td>-0.26</td>
<td>3039</td>
<td>3057</td>
<td>304</td>
<td>737</td>
</tr>
<tr>
<td>200</td>
<td>50</td>
<td>1906.696</td>
<td>1904.189</td>
<td>0.13</td>
<td>28.977</td>
<td>28.678</td>
<td>1.03</td>
<td>1144</td>
<td>766</td>
<td>65</td>
<td>351</td>
</tr>
<tr>
<td>200</td>
<td>100</td>
<td>1882.793</td>
<td>1877.641</td>
<td>0.27</td>
<td>29.277</td>
<td>30.052</td>
<td>-2.65</td>
<td>4104</td>
<td>3149</td>
<td>359</td>
<td>895</td>
</tr>
<tr>
<td>200</td>
<td>200</td>
<td>1873.964</td>
<td>1871.614</td>
<td>0.13</td>
<td>29.276</td>
<td>29.343</td>
<td>-0.23</td>
<td>10811</td>
<td>9672</td>
<td>690</td>
<td>1539</td>
</tr>
</tbody>
</table>

3.3 Large numbers of draws

We want to investigate the effects that a larger number of simulation draws has on the gap. This is interesting, among other reasons, because we can expect more complex advanced discrete choice models to require a higher number of draws to be accurately simulated (Pacheco, 2020).

We solve the same problem instance as in the previous subsection, this time with up to $R = 1000$ draws. The number of participants has been kept at $N = 50$, in order to guarantee that the MILP still finds a feasible solution. The results (for the MSLE problem) can be seen in Table 5.

It appears that a higher number of draws indeed has the effect of reducing the optimality gap, however, it has to be taken into account that very often the MILP will not converge in the given time limit of three hours for such large instances, making it more likely that we compare our output to solutions that are actually sub-optimal. On the other hand, by far the biggest part of computational time in the MILP is spent proving the lowerbound, i.e. optimality, meaning that the objective value of the solution often is already optimal, even if at the time limit there is a nonzero gap to be observed. Furthermore, for 800 draws and more, the MILP does not manage to find a feasible solution in three hours, indicating that our approach could be applied as a heuristic to find good feasible solution fast in large scale scenarios, which could then be used as...
warm start solutions for the MILP.

Table 5: Comparing our decomposition method with the full MILP in terms of $sLL$ and runtime for large numbers of draws ($N = \text{population size}, R = \text{number of draws}, sLL = \text{simulated Log-Likelihood}, M = \text{MILP}, D = \text{decomposition}, T = \text{time in sec.}$)

<table>
<thead>
<tr>
<th>$N$</th>
<th>$R$</th>
<th>$sLL-M$</th>
<th>$sLL-D$</th>
<th>Gap [%]</th>
<th>$T-M$</th>
<th>$T-D$</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>20</td>
<td>-29.417</td>
<td>-29.908</td>
<td>1.67</td>
<td>22</td>
<td>6</td>
</tr>
<tr>
<td>50</td>
<td>50</td>
<td>-29.294</td>
<td>-31.173</td>
<td>6.41</td>
<td>279</td>
<td>26</td>
</tr>
<tr>
<td>50</td>
<td>100</td>
<td>-28.885</td>
<td>-29.42</td>
<td>1.85</td>
<td>1375</td>
<td>42</td>
</tr>
<tr>
<td>50</td>
<td>150</td>
<td>-29.973</td>
<td>-30.092</td>
<td>0.4</td>
<td>2852</td>
<td>70</td>
</tr>
<tr>
<td>50</td>
<td>200</td>
<td>-30.091</td>
<td>-30.101</td>
<td>0.03</td>
<td>10800</td>
<td>131</td>
</tr>
<tr>
<td>50</td>
<td>250</td>
<td>-30.741</td>
<td>-30.775</td>
<td>0.11</td>
<td>10800</td>
<td>156</td>
</tr>
<tr>
<td>50</td>
<td>300</td>
<td>-30.837</td>
<td>-30.843</td>
<td>0.02</td>
<td>10800</td>
<td>133</td>
</tr>
<tr>
<td>50</td>
<td>400</td>
<td>-30.632</td>
<td>-30.638</td>
<td>0.02</td>
<td>10800</td>
<td>130</td>
</tr>
<tr>
<td>50</td>
<td>600</td>
<td>-30.479</td>
<td>-30.51</td>
<td>0.1</td>
<td>10800</td>
<td>289</td>
</tr>
<tr>
<td>50</td>
<td>800</td>
<td>-32.035</td>
<td></td>
<td></td>
<td>10800</td>
<td>319</td>
</tr>
<tr>
<td>50</td>
<td>1000</td>
<td>-30.523</td>
<td></td>
<td></td>
<td>10800</td>
<td>349</td>
</tr>
</tbody>
</table>

3.4 Decomposition as a warm start

Another interesting perspective is to investigate the effectiveness of using our fast, yet slightly suboptimal decomposition algorithm to generate a good warm start solution for the MILP. This allows to still speed up computation, while maintaining guaranteed optimality. The results (for the MSLE problem) can be seen in Table 6. We observe varied outcomes: For smaller instances ($N \leq 100$) most instances show a speed-up of around 50%, however, outliers exist where the warm start solution influences the computational time negatively. For larger instances, in almost all cases the warm start solution does not seem to lead to a significant speed-up. Interesting to observe is also that we can have slightly better or worse solutions when solving the MILP with a warm start versus solving it without, even though the Branch & Bound algorithm used by Gurobi to solve the MILP guarantees optimality in both cases. This is an indication that the problem might be highly susceptible to numerical issues. This is insofar illuminating that the gaps we achieve with our decomposition method might be small enough to be ignored, especially with higher numbers of draws, as seen in subsection 3.3. In general the utility of using our method as a warm start solution is not clearly ascertainable, and thus requires further investigation.
Table 6: Comparing our decomposition method as a warm start with the full MILP in terms of sLL and runtime (N = population size, R = number of draws, sLL = simulated Log-Likelihood, M = MILP, D = decomposition, M+ = MILP with warm start, T = time in sec.)

<table>
<thead>
<tr>
<th>N</th>
<th>R</th>
<th>sLL-M</th>
<th>sLL-D</th>
<th>Gap [%]</th>
<th>sLL-M+</th>
<th>Gap[%]</th>
<th>T-M</th>
<th>T-D</th>
<th>T-M+</th>
<th>rel. Speed-up [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>50</td>
<td>-29.685</td>
<td>-29.842</td>
<td>-0.53</td>
<td>-29.685</td>
<td>0</td>
<td>578</td>
<td>38</td>
<td>290</td>
<td>49.87</td>
</tr>
<tr>
<td>50</td>
<td>100</td>
<td>-29.956</td>
<td>-30.954</td>
<td>-3.33</td>
<td>-29.956</td>
<td>0</td>
<td>2444</td>
<td>70</td>
<td>1308</td>
<td>46.48</td>
</tr>
<tr>
<td>50</td>
<td>150</td>
<td>-30.295</td>
<td>-30.356</td>
<td>-0.2</td>
<td>-30.295</td>
<td>0</td>
<td>1759</td>
<td>106</td>
<td>1846</td>
<td>-4.95</td>
</tr>
<tr>
<td>50</td>
<td>200</td>
<td>-30.352</td>
<td>-30.414</td>
<td>-0.2</td>
<td>-30.352</td>
<td>0</td>
<td>5074</td>
<td>131</td>
<td>10800</td>
<td>-112.86</td>
</tr>
<tr>
<td>100</td>
<td>50</td>
<td>-64.812</td>
<td>-65.555</td>
<td>-1.15</td>
<td>-64.886</td>
<td>-0.11</td>
<td>4181</td>
<td>60</td>
<td>630</td>
<td>84.93</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>-63.995</td>
<td>-64.118</td>
<td>-0.19</td>
<td>-63.918</td>
<td>0.12</td>
<td>10800</td>
<td>177</td>
<td>3890</td>
<td>63.98</td>
</tr>
<tr>
<td>100</td>
<td>150</td>
<td>-64.524</td>
<td>-64.551</td>
<td>-0.04</td>
<td>-64.524</td>
<td>0</td>
<td>10800</td>
<td>379</td>
<td>5184</td>
<td>52</td>
</tr>
<tr>
<td>100</td>
<td>200</td>
<td>-65.669</td>
<td>-66.48</td>
<td>-1.23</td>
<td>-65.309</td>
<td>0.55</td>
<td>10800</td>
<td>122</td>
<td>10800</td>
<td>0</td>
</tr>
<tr>
<td>200</td>
<td>50</td>
<td>-124.5</td>
<td>-124.5</td>
<td>0</td>
<td>-125.343</td>
<td>-0.68</td>
<td>6341</td>
<td>157</td>
<td>2988</td>
<td>52.88</td>
</tr>
<tr>
<td>200</td>
<td>100</td>
<td>-122.97</td>
<td>-122.823</td>
<td>0.12</td>
<td>-131.467</td>
<td>-6.91</td>
<td>10800</td>
<td>319</td>
<td>10800</td>
<td>0</td>
</tr>
<tr>
<td>200</td>
<td>150</td>
<td>-124.1</td>
<td>-123.973</td>
<td>0.11</td>
<td>-124.104</td>
<td>0</td>
<td>10800</td>
<td>464</td>
<td>10800</td>
<td>0</td>
</tr>
<tr>
<td>200</td>
<td>200</td>
<td>-124.34</td>
<td>-123.786</td>
<td>0.45</td>
<td>-124.468</td>
<td>-0.1</td>
<td>10800</td>
<td>467</td>
<td>10800</td>
<td>0</td>
</tr>
</tbody>
</table>
4 Plan for future research

4.1 Modeling framework

This section formally introduces the groundwork on the methodology that is going to be investigated in future research.

Choice based optimization problems like MSLE complicate the direct application of decomposition methods due to an inherent non-linearity of the constraints modeling the choice of the alternative with the highest utility. There are multiple ways to tackle non-linearity in MILPs, for example with big-M reformulations, piece-wise linearization or the modeling of the problem as a (mixed-integer) convex quadratic program. An additional, original, linearization technique has been developed by the authors, however, it is more appropriate to characterize it as a "quasi"-linearization, as the product of two variables is defined with the use of fixed constants, depending on the variable value. It only makes sense to use the quasi linearization technique in the context of knapsack Benders (and extensions thereof). We furthermore give a brief summary on column generation, as it as promising alternative to Benders decomposition. Table 7 below shows the compatibility of different decomposition methods with various linearization techniques.

Table 7: Compatibility between decomposition methods and linearization methods

<table>
<thead>
<tr>
<th></th>
<th>Big-M</th>
<th>Quasi</th>
<th>Piece-wise</th>
<th>MICQP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benders (knapsack)</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
<td>?</td>
</tr>
<tr>
<td>Benders (standard)</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Column generation</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Benders + Col. gen.</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

It is important to note that there are possible issues arising from the piece-wise approximation of products involving relaxed integral variables in the context of total unimodularity. Furthermore it is difficult to predict the compatibility of knapsack Benders and MICQP, as no previous theory exists to rely on. What follows are short descriptions of the methods that are to be explored.
4.2 Decomposition methods

**Column generation** Combinatorial optimization problems featuring complicating constraints can be addressed by a column generation approach (Desaulniers et al., 2006). Such an approach iteratively adds the decision variables involved in the complicating constraints, employing duality theory to determine whether there are more decision variables that can be added (i.e. enter the basis) and provide an improved solution (i.e. they have negative reduced costs). This is achieved by solving a pricing sub-problem, in which the objective function is the reduced cost of the new candidate variable subject to problem-specific constraints. After solving the pricing sub-problems, if there are multiple variables that can potentially enter the basis, the variable exhibiting the most negative reduced cost can be added. Instead, if there are no variables that can be added to the basis, an optimal solution has been found and the algorithm can be terminated at an early stage. A flowchart describing the several steps for column generation is shown in Figure 6. Column generation is especially interesting for choice-based problems, as many of the complicating choice variables are set to zero, while only the variable corresponding to the choice with the highest utility is set to be equal to one. This means all other choice variables will remain non-basic and can technically be excluded. It is furthermore possible, that the performance of a column generation approach could further be improved by grouping certain variables together, instead of treating them all individually. A similar "smoothing" approach showed to improve the performance of for example Lagrangean decomposition in the same context, see Pacheco et al. (2018). The extension of column generation methods to mixed-integer convex quadratic problems is often referred to as Simplicial Decomposition (SD) or the Frank-Wolfe algorithm (see Bettiol, 2019). For a review on column generation, we refer to Lübbecke (2010).

Figure 6: Flowchart for column generation
4.3 Linearization methods

In this subsection we briefly describe the different methods anticipated to be capable of linearizing the nonlinear choice-constraints, i.e.

\[ H_{nr} = \sum_i U_{inr} \omega_{inr} \]  

where \( H_{nr} \) represents the highest utility and \( \omega \) the choice variables for every alternative \( i \), individual \( n \), and scenario \( r \).

**big-M linearization**  This is the most standard way of dealing with nonlinear constraints as the ones in (*) . It gets its name from utilizing a large enough constant \( M \) to deactivate constraints if needed. The nonlinear product is modeled with an auxiliary variable \( \eta_{inr} = U_{inr} \omega_{inr} \):

\[
\begin{align*}
H_{nr} &= \sum_i \eta_{inr} \\
\eta_{inr} &\leq \omega_{inr} M \\
\eta_{inr} &\geq -\omega_{inr} M \\
\eta_{inr} &\leq U_{inr} + (1 - \omega_{inr})M \\
\eta_{inr} &\geq U_{inr} - (1 - \omega_{inr})M
\end{align*}
\]

This linearization method only works if \( \omega \) is kept integral, thus making it incompatible with knapsack Benders.

**Quasi linearization**  This linearization method only works in the context of knapsack Benders, where in the subproblem, the parameter to be optimized is fixed, and thus the utilities are as well. However, in order to derive Benders cuts, the fixing of variables to a given value has to happen in specific way, with the fixed value isolated on the rhs of the constraint. Technically, the fixed value should also not appear in the coefficient matrix, but so far we have not been able to come up with a reformulation that achieves this. The current approach, implemented for the maximum simulated likelihood estimation problem, where the parameter to be estimated is called \( \beta_k \), is shown below. Note that, instead of modelling the product \( U_{inr} \omega_{inr} \) we instead model the \( k \) products \( \eta_{inrk} = \beta_k \omega_{inr} \), where \( k \) is the dimension of the optimization variable.

\[
\begin{align*}
\chi_{inr} + \omega_{inr} &= 1 \\
\eta_{inrk} + \beta_k^{\text{fixed}} \chi_{inr} &= \beta_k^{\text{fixed}} \\
\sum_i \eta_{inrk} &= \beta_k
\end{align*}
\]
**Piece-wise linearization**  For this method, we should first write the product $\beta_k \omega_{inr}$ in terms of quadratic expressions: we define two new variables $\eta_{inrk}$ and $\phi_{inrk}$ in the following way:

\[
\eta_{inrk} = \frac{1}{2} (\beta_k + \omega_{inr}) \\
\phi_{inrk} = \frac{1}{2} (\beta_k - \omega_{inr})
\]

Now it holds that:

\[
\beta_k \omega_{inr} = \eta_{inrk}^2 - \phi_{inrk}^2
\]

We then need to come up with good bounds on $\beta_k \in [lb_k, ub_k]$, as for $\omega$ these bounds are already given by $[0,1]$. Then the classical way of piece-wise linearizing would be to first divide $[lb_k, ub_k]$ into $m$ separate segments and define $a_j$, $(j \in \{0,1,\ldots,m\})$ as the breakpoints, $a_0 < a_1 < \cdots < a_m$. Then, we can approximately linearize the non-linear function $f(x) = \bar{x}$ for $x \in [a_0,a_m]$ as follows:

\[
\sum_j \lambda_j a_j = x \\
\sum_j \lambda_j f(a_j) = \bar{x} \\
\sum_j \lambda_j = 1 \\
\lambda \geq 0
\]

with the restriction that only two adjacent $\lambda_j$’s are allowed to be nonzero. This is usually modeled using SOS2 constraints, however, this requires the use of $m + 1$ binary variables, which, in the context of relaxed choice variables $\omega$, would make little sense. However, in some scenarios these adjacency constraints can be redundant, for example if the function to be linearized is either convex and minimized by the objective, or concave and maximized (see [Aimms, 2016]). It is thus open to investigation if the approach can be made to work in a relaxation setting. Another approach worth investigating is to employ a method developed by [Li and Yu, 1999], which uses the fact that, if the slope of $f$ is non-decreasing, which is the case for $f(x) = x^2$, its possible to write a piece-wise linearization using absolute values, without having to introduce auxiliary binary variables to deal with non-convex parts. First, a univariate mathematical function is formulated via a piece-wise linear sum of absolute expressions. Denote $s_j$, $(j \in \{0,1,\ldots,m-1\})$ as the slopes of each line between $a_j$ and $a_{j+1}$ computed using the following equation:

\[
s_j = \frac{f(a_{j+1}) - f(a_j)}{a_{j+1} - a_j}, \quad \forall j \in \{0,\ldots,m-1\}
\]
An equivalent piece-wise linear form of non-linear function $f(x)$ can then be reformulated as follows:

$$L(f(x)) = f(a_0) + s_0(x - a_0) + \sum_{j=1}^{m-1} \frac{s_j - s_{j-1}}{2} (|x - a_j| + x - a_j)$$

where the absolute expressions can be linearized as follows:

$$L(f(x)) = f(a_0) + s_0(x - a_0) + \sum_{j=1}^{m-1} (s_j - s_{j-1}) (\sum_{h=0}^{j-1} d_h + x - a_j)$$

$$x + \sum_{j=0}^{m-2} d_j \geq a_{m-1}$$

$$d_j \leq a_{j+1} - a_j \quad \forall j \in \{0, \ldots, m - 1\}$$

$$d \geq 0$$

The overall advantage of a piece-wise linear approximation is that the accuracy of the linearization can be controlled using the number of breakpoints $a_j$, thus a clear trade-off between accuracy and efficiency is given. A clear downside is that it is very unclear whether the beneficial and crucial properties of the knapsack reduction still hold when the products with the integral variables are approximated. Positive in this regard is the fact that the approximation is most crucial at 0, where it can be made equal to the function by adding a breakpoint. Furthermore, there is the need for valid bounds on the parameters $\beta$ to be estimated, however, if we manage to employ this method successfully in a relaxation context, adding more breakpoints simply means adding more continuous variables to a linear program, which is not expensive. This allows to approximate the square function over a large interval with high accuracy.

**MICQP formulation**  Another way of dealing with nonlinearity in an MILP would be switch the paradigm completely and regard the problem as mixed-integer convex quadratic program. It is yet to be shown that choice-based optimization programs can be written in a convex way, and it might well depend on problems specifics. The intuition to why it might work is the following: since we can always shift the utilities without changing the optimal solution, we can wlog assume all utilities are negative. This means that in the product $U_{inr} \omega_{inr}$, again lets say in a relaxation setting where both variables are continuous, we are multiplying a negative continuous variable with a positive continuous variable, plus the constraint (**) can be wlog written as

$$H_{nr} \leq \sum_i U_{inr} \omega_{inr}$$
since the lower bound is given by the constraints

\[ H_{nr} \geq U_{inr}, \forall i \]

This means we have a variable \( H_{nr} \) which is trying to stay below a sum of concave curves, which implies that the feasible region for this constraint is, in theory, convex.

5 Conclusion

In this paper, we develop a mixed integer linear program (MILP) for the simulated maximum likelihood estimation (MLSE) problem and construct a Benders decomposition approach to speed up the solution process. The methodology can be applied to any advanced discrete choice model and makes use of total unimodularity to keep the master problem linear in the decomposition, avoiding the typical bottleneck in efficiency for a Benders decomposition. The results on a binary logit discrete choice model show an average speed up of factor 35, with instances being solved up to 100 times faster. Small deviations in the optimal solution values between decomposition and full MILP are present. This issue will be tackled in future research by investigating different ways of dealing with non-linearity, before moving on to alternative decomposition approaches.

6 References


24