The non-parametric/machine learning future?

Chris Conlon

Fall 2026

NYU Stern

Nonparametrics?

What do you mean by non-parametric?

Mostly we mean putting a flexible distribution on $f(\beta_i, \alpha_i \mid \theta)$ (and keeping logit error on ε_{ij})

$$u_{ij} = \beta_i x_j - \alpha_i p_j + \xi_j + \varepsilon_{ij} \text{ with } f(\beta_i, \alpha_i \mid \theta)$$

- Fixed Grids (Fox, Kim, Ryan, Bajari 2011, Heiss, Hetzenecker, Osterhaus 2022, Nevo Turner Williams 2016): draw from "prior" of β_i, compute $\sigma_{ij}(\beta_i)$ and choose weights on each i, π_i.
- Compiani (QE 2022): approximate $\sigma_j^{-1}(\mathcal{S}_t, \mathbf{x}_t^{(2)})$ directly with Bernstein Polynomials (ditches $\varepsilon \to$ very hard)
- Ao Wang (JE 2022): use polynomial sieves: $\mathbb{E}\left[\left(\sigma_j^{-1}\left(\mathcal{S}_t; \mathbf{x}_t^{(2)}, F\right) - X_t^{(1)}\beta^{(1)}\right)\phi_k\left(Z_{jt}\right)\right] = 0$
- Lu, Shi, Tao (JE 2023): use partially linear model: $\log\left(s_{jt}/s_{0t}\right) = X_{1,jt}'\beta^0 + \psi^0\left(X_{2,jt}; IV_{J,t}\right) + \xi_{jt}$ where $\psi^0\left(x_{2,jt}; IV_{J,t}\right) = \log\left[\frac{\int \frac{\exp\left(x_{2,jt}'v\right)}{\exp\left(IV_{J,t}(v)\right)} f^0(v)dv}{\int \frac{1}{\exp\left(IV_{J,t}(v)\right)} f^0(v)dv}\right]$.

But these are still only as good as characteristics.

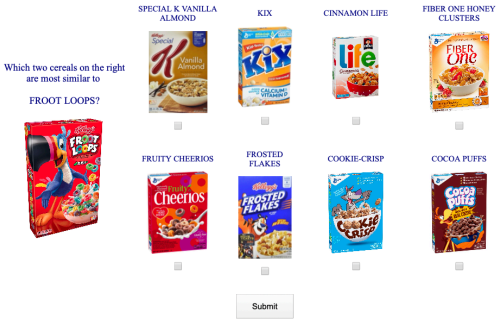
- **Goal:** Assign a low-dimensional vector of characteristics $x_j \in \mathbb{R}^m$ for each product using triplets of the form "j is more similar to k than to ℓ".
- The "t" in t-STE comes from using a **Student-t kernel** to model distances—giving heavy tails and better separation of clusters.

$$\max_{\mathbf{x}} \sum_{(j,k,\ell)\in\mathcal{T}} \ln\left(\pi_{jk\ell}\right) \quad \text{where} \quad \pi_{jk\ell} = \frac{\left(1+\frac{\left\|x_j-x_\ell\right\|^2}{\alpha}\right)^{-\frac{\alpha+1}{2}}}{\left(1+\frac{\left\|x_j-x_k\right\|^2}{\alpha}\right)^{-\frac{\alpha+1}{2}}+\left(1+\frac{\left\|x_j-x_\ell\right\|^2}{\alpha}\right)^{-\frac{\alpha+1}{2}}}$$

- This is basically maximium (log) likelihood for a binary outcome model (closer to ℓ than k).
  - Fit with canned routine (which does gradient descent).
  - α is a (somewhat arbitrary) tuning parameter.

Unobserved Characteristics: Magnolfi Maclure Sorensen (2023)

FIGURE 1: Sample survey page

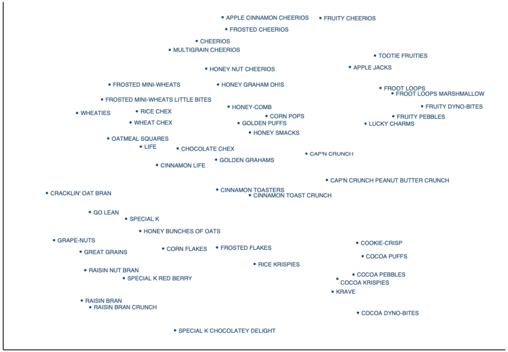


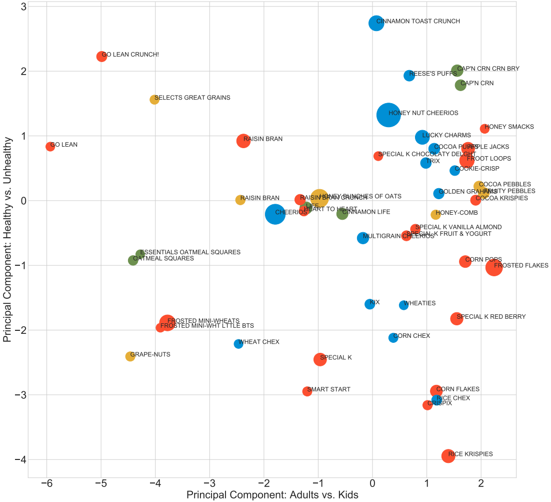
What if we could first estimate unobserved characteristics?

- Is j more similar to k or l?
- Get a $m \times J$ matrix with m factors (embeddings).
- Idea: m is small (like 3-4).
- Use these as characteristics in BLP demand model.

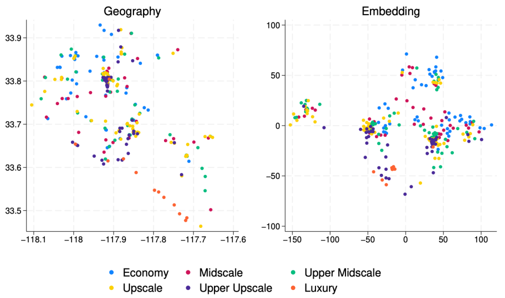
Unobserved Characteristics: Magnolfi McClure Sorensen (2023)

FIGURE 2: Plot of two-dimensional embedding

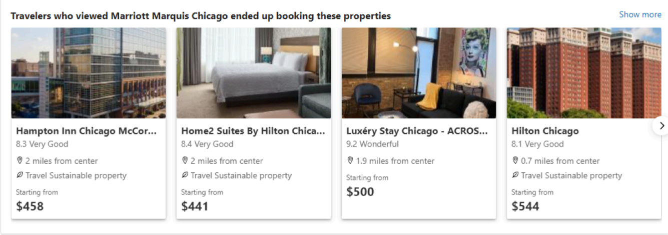




Reverse Engineering Hotel Recommendations: McClure (2025)

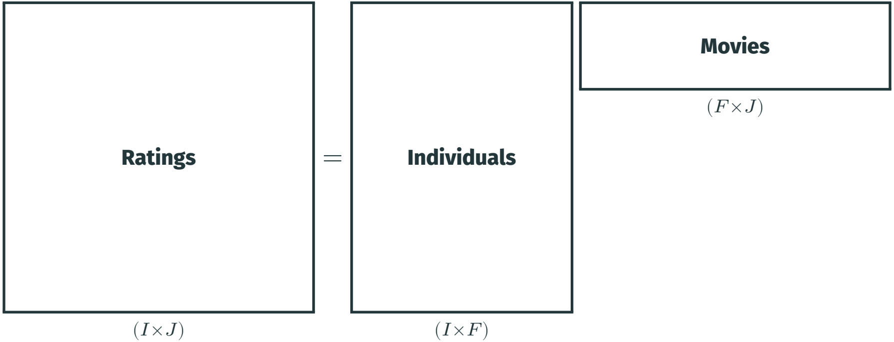


Appendix Figure 1: Recommendations at Booking.com



- Scrape observed "you may also like" recommendations from hotels and use those to construct triplets
- Feed the triplets into tSTE embedding model to get a matrix **X**: $(J \times F)$ where F is small (2-3).
- In both cases plug these in as characteristics to demand model (no guarantee they explain substitution patterns).

Idea #2: Low Rank Matrix Factorization: aka Netflix Prize



- Even if Ratings are sparse, we fit the observed cells and predict the rest!
- Idea: Approximate with a low rank (F) factor model (such as SVD)
- Maybe embeddings are reverse-engineering “collaborative filter”.

Conlon, Mortimer, Sarkis (2024): Structural Low Rank Approximations

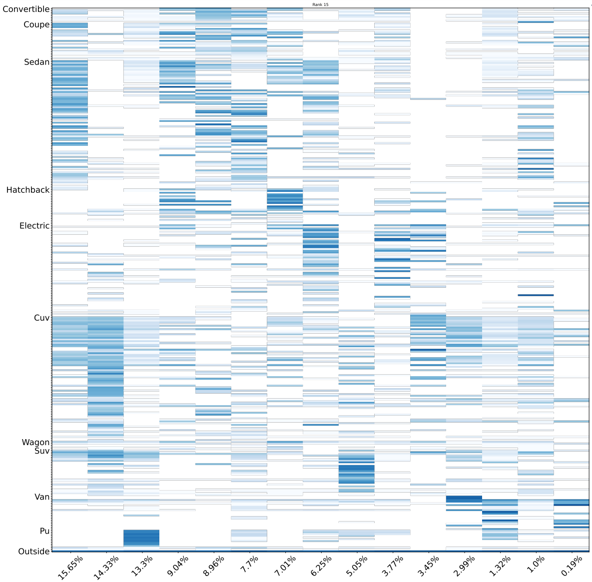
Fix rank$(\mathbf{D}(\mathbf{S},\pi)) = I$, and for each choice of I solve:

$$\min_{(\mathbf{S},\pi)\geq 0} \left\|\mathcal{P}_\Omega(\mathcal{D} - \mathbf{D}(\mathbf{S},\pi))\right\|_{\ell_2} + \lambda \left\|\mathcal{S} - \mathbf{S}\,\pi\right\|_{\ell_2} \text{ with } \|\pi\|_{\ell_1} \leq 1, \quad \|\mathbf{s_i}\|_{\ell_1} \leq 1.$$

- Goal: estimate $\mathbf{s_i}$ (choice probabilities) and corresponding weights π_i (Finite Mixture) in product space
  - Consistent with $U_{ij} = V_{ij} + \varepsilon_{ij}$ and logit error.
- Constraints: Choice probabilities s_{ij} sum to one, type weights π_i sum to one.
  - ℓ_1 constraints lead to sparsity.
- Idea: Control the rank by limiting I directly
  - Use cross validation to select # of types I and Lagrange multiplier λ.
- Matrix completion: We can construct estimates of $\mathbf{D}(\mathbf{S},\pi)$ including elements of $\mathcal{P}_{\overline{\Omega}}$.

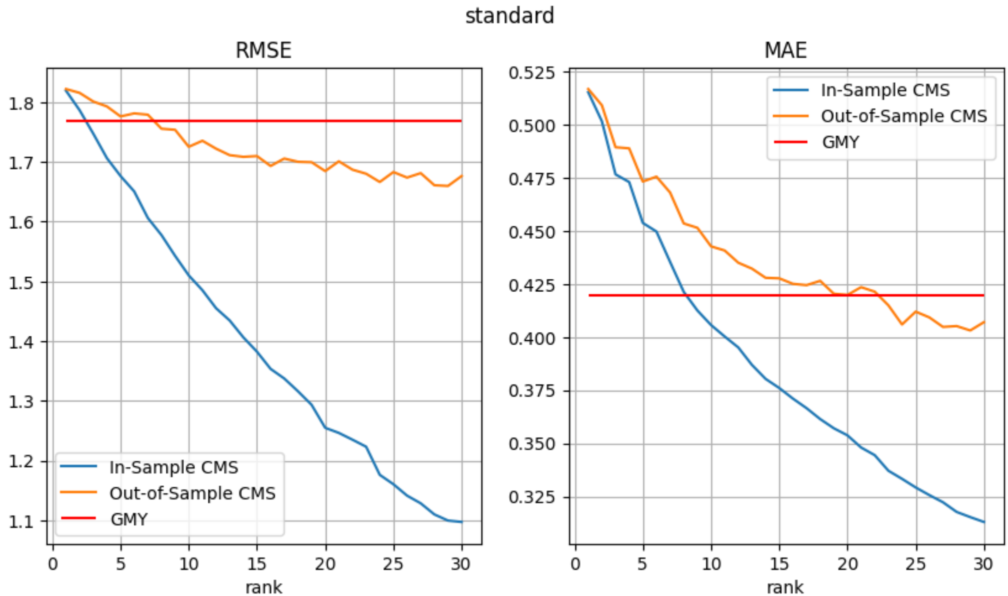
$$\mathbf{D} = \frac{1}{\mathbf{s}} \sum_{i=1}^{I} \pi_i\,\mathbf{s_i} \times \left[\frac{\mathbf{s_i}}{1-\mathbf{s_i}}\right]^T$$

Profiles of Types (Rank 15)



- Each column of the matrix $\mathbf{S}$ $(I \times J)$ represents a "type" $\mathbf{s_i}$ (a $J \times 1$) vector of choice probabilities.
- We see both the sparsity as well as specific consumer segments (or "types").
- We also obtain the share of each type in the population π_i.
- Remember that $u_{ij} - u_{i0} = \log s_{ij} - \log s_{i0}$, so that we identify indirect utilities (relative to outside good).
- We still need to estimate $\frac{\partial u_{ij}}{\partial p_j}$ somehow.

In-Sample Performance

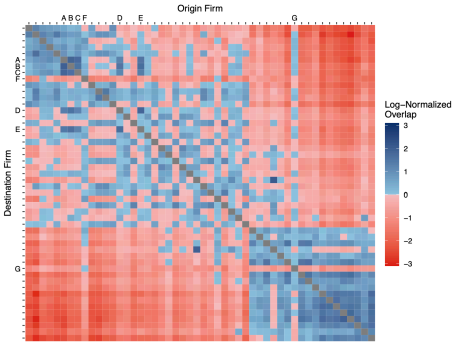


Top Substitutes: Ford F-Series

Model	Raw	Logit	CMS I=15	CMS I=30	GMY
Ram Pickup	24.59	0.88	21.46	22.23	19.4
Gmc Sierra	20.29	0.61	14.97	21.92	17.27
Chevrolet Silverado	15.62	0.78	13.408	19.63	33.62
Toyota Tundra	12.98	0.55	16.32	12.79	2.29
Toyota Tacoma	6.31	0.76	3.39	3.13	2.83
Chevrolet Colorado	4.64	0.63	3.22	2.86	2.87
Gmc Canyon	2.3	0.3	0.76	1.38	1.02
Nissan Frontier	1.63	0.43	0.92	1.69	0.61
Jeep Wrangler	1.59	0.69	1.33	0.94	0.06
Nissan Titan	0.7	0.05	1.18	1.17	0.18
Ford Explorer	0.63	0.38	0.16	0.14	0.71

Idea #3: Customer Overlap (Einav, Guido, Klenow 2025)

Figure 4: Hotel Overlap



Note: This picture visualizes all pairwise log-normalized overlap measures for the top 50 hotel chains in the U.S. Hotel chains sorted from highest non-hotel spending rank to lowest non-hotel spending rank across both axes, going from left to right and from top to bottom (i.e. highest ranked chains are in the top left and lowest ranked chains are in the bottom right). A single cell of this matrix can be interpreted as the log-normalized overlap of the column chain with the row chain (i.e. $C_{col \to row}$). Thus, any column can be interpreted as how much that chain's customers relatively overlap with other chains. Similarly, any row can be interpreted as how much all other chains' customers overlap with the row chain. Values above zero are colored blue, with darker colors indicating higher log-normalized overlap. Similarly, values below zero are colored red, with darker shades

- Encode a $J \times 1$ vector $\mathbf{q_i}$ of every purchase (0/1) within the category for the year
- They use credit card data (don't see items, only stores).
- Could use Nielsen panelists.
- How many customers do two stores have in common? $C_{j \to k} = \frac{\sum_{i \in \mathcal{C}_j} q_{i,k}}{\sum_{i \in \mathcal{C}_j} q_{i,(-j)}}$.
- Use to estimate diversion ratios (but not a demand model).
- Conlon Rao (JPE, forthcoming) use a similar measure to estimate nesting parameter.
- Atalay et. al (JPE, 2024) use a similar measure to assign products to nests.

Thanks!