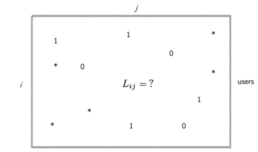
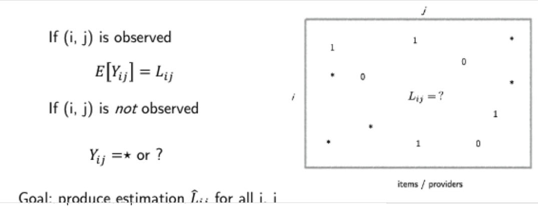
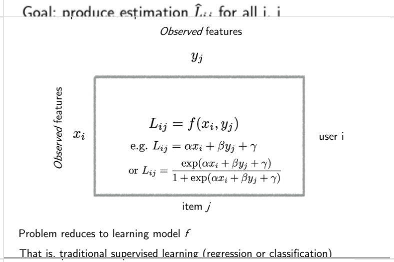
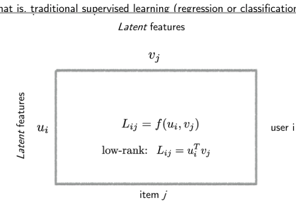
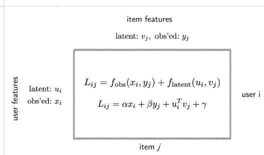
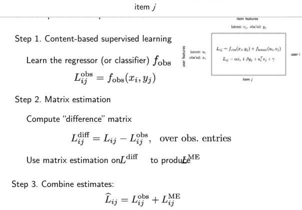
Matrix Estimation Meets Content Based Filtering

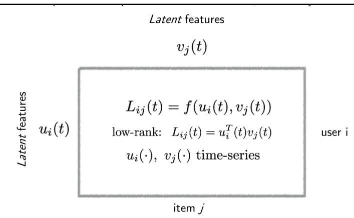
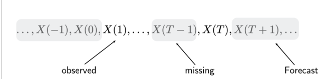
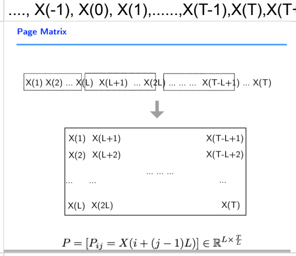
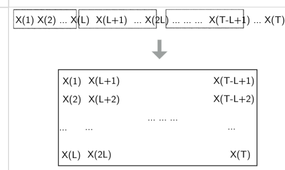
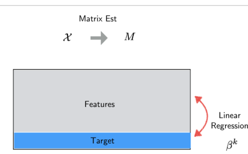
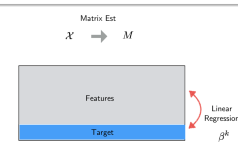
Description	What	Limitations/Assumptions	Example/Formula	Reference/Comments
Outline	Matrix Estimation meets Content-based Matrix Estimation across time Matrix Estimation everything together			
Combined Approach	1. Content based supervised learning 2. Matrix Estimation 3. Combine estimates			

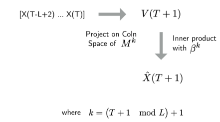
Matrix Estimation Meets Content Based Filtering

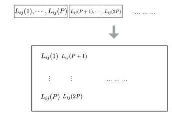
Description	What	Limitations/Assumptions	Example/Formula	Reference/Comments
Problem statement Prediction problem: complete the matrix				
Problem statement Prediction problem: complete the matrix		Observations : Y_{ij} over **i** in users, **j** in items Goal: produce estimation L^ij for all **i, j**		
Problem statement Prediction problem: complete the matrix Content-based: Model		Problem reduces to learning model f That is, traditional supervised learning (regression or classification)		
Problem statement Prediction problem: complete the matrix Matrix Estimation: Model		Problem reduces to learning "factorization" of the matrix either through similarites or algebraic approaches		
Matrix Estimation meets Content-based: Model Prediction problem: complete the matrix		Problem reduces to learning model f That is, traditional supervised learning (regression or classification)		
Matrix Estimation meets Content-based: Algorithm		1. Content based supervised learning 2. Matrix estimation 3. Combine estimates		

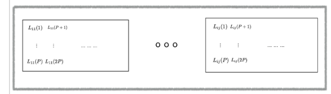
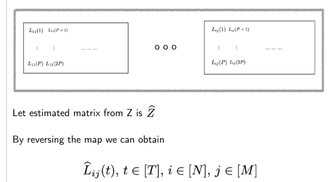
Matrix Estimation Across Time

Description	What	Limitations/Assumptions	Example/Formula	Reference/Comments

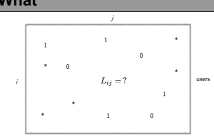
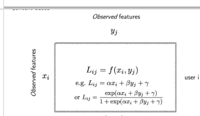
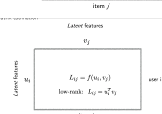
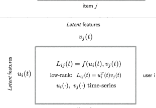
Matrix Estimation over time: Model Prediction problem: complete the matrix over time, indexed by t		Problem reduces estimating time varying matrix where Latent factors are time varying, observations are partial and noisy		
Matrix Estimation over TIme: Model Prediction problem: complete the matrix over time, indexed by t	Multiple time series	$L_{ij}(t),\ i \in [N],\ j \in [M]$		
Matrix Estimation over TIme: Model Prediction problem: complete the matrix over time, indexed by t	They obey latent structure where each component of u_i , v_j is a structured time-series	$L_{ij}(t) = u_i(t)^T v_j(t),\ \ u_i(t),\ v_j(t) \in \mathbb{R}^d$		
Matrix Estimation over TIme: Model Prediction problem: complete the matrix over time, indexed by t	Observations	Partial, noisy observations of the $L_{ij}(t),\ i \in [N],\ j \in [M]$ NOT $u_i(t),\ v_j(t) \in \mathbb{R}^d$		
A digression: time series imputation, forecasting	Ground truth of interest: ...., X(-1), X(0), X(1),......,X(T-1),X(T),X(T+1),.....	Observations: 		
Page Matrix				
Imputation of time series data	Transform to Matrix, **Do Matrix Estimation**, Undo Transformation			
Forecasting of time series data	Transform to Matrix, **Do Matrix Estimation**, Regression , Prediction			
Forecasting of time series data	Transform to Matrix, Do Matrix Estimation, **Regression** , Prediction			

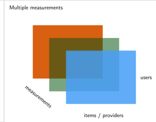
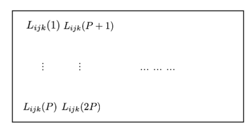
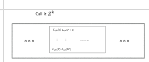
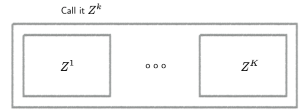
<table>
<tr><td>Forecasting of time series data</td><td>Transform to Matrix, Do Matrix Estimation, Regression , Prediction</td><td>
</td><td></td><td></td></tr>
<tr><td>Does it really work</td><td>
<table>
<tr><th></th><th colspan="4">Mean Imputation (NRMSE)</th><th colspan="4">Mean Forecasting (NRMSE)</th></tr>
<tr><th></th><th>Electricity</th><th>Traffic</th><th>Synthetic</th><th>Financial</th><th>Electricity</th><th>Traffic</th><th>Synthetic</th><th>Financial</th></tr>
<tr><td>mSSA</td><td>0.391</td><td>0.494</td><td>0.253</td><td>0.283</td><td>0.483</td><td>0.525</td><td>0.196</td><td>0.358</td></tr>
<tr><td>SSA</td><td>0.519</td><td>0.608</td><td>0.626</td><td>0.466</td><td>0.552</td><td>0.704</td><td>0.522</td><td>0.592</td></tr>
<tr><td>LSTM</td><td>NA</td><td>NA</td><td>NA</td><td>NA</td><td>0.551</td><td>0.473</td><td>0.444</td><td>1.203</td></tr>
<tr><td>DeepAR</td><td>NA</td><td>NA</td><td>NA</td><td>NA</td><td>0.484</td><td>0.474</td><td>0.331</td><td>0.395</td></tr>
<tr><td>TRMF</td><td>0.694</td><td>0.512</td><td>0.325</td><td>0.513</td><td>0.534</td><td>0.570</td><td>0.267</td><td>0.464</td></tr>
<tr><td>Prophet</td><td>NA</td><td>NA</td><td>NA</td><td>NA</td><td>0.582</td><td>0.617</td><td>1.005</td><td>1.296</td></tr>
</table>
</td><td>Out-performs:
DeepAR from Amazon
Prophet from Facebook
LSTM, the shiny neural network approach</td><td>Easy to use implementation:
tspdb.mit.edu

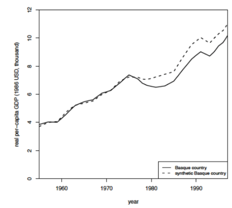
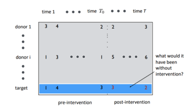
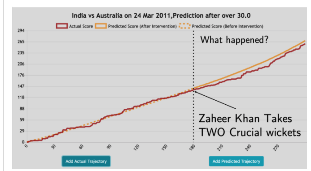
NeurIPS 2020 demo:
https://www.powtoon.com/s/fkVh3axA4Jy/1/m</td><td></td></tr>
<tr><td>Matrices Changing with Time</td><td>1. Convert time series for each component into a (Page) matrix</td><td></td><td></td><td></td></tr>
<tr><td>Matrices Changing with Time</td><td>2. Concatenate matrices of all components along columns observation over of a T time units

This combine matrix, say Z, has
P rows
T/P x N x M columns</td><td></td><td></td><td></td></tr>
<tr><td>Matrices Changing with Time</td><td>3. Perform matrix estimation over Z</td><td>
</td><td></td><td></td></tr>
</table>

Everything together

<table>
<tr><th>Description</th><th>What</th><th>Limitations/Assumptions</th><th>Example/Formula</th><th>Reference/Comments</th></tr>
<tr><td>Recommendation: Problem statement
Prediction problem: complete the matrix</td><td>
</td><td></td><td></td><td></td></tr>
<tr><td>Content based - Model 1</td><td>
</td><td>Problem reduces to learning model f
That is, traditional supervised learning (regression or classification)</td><td></td><td></td></tr>
<tr><td>Matrix Estimation - Model 2</td><td>
</td><td>Problem reduces to learning "factorization" of the matrix either through similarities or algebraic approaches</td><td></td><td></td></tr>
<tr><td>Matrix Estimation over time - Model 3</td><td>
</td><td>Problem reduces estimating time varying matrix where
Latent factors are time varying ,
observations are partial and noisy</td><td></td><td></td></tr>
</table>

Multiple measurements				
Recommendation: Model Everything	Put everything together Users: N, Items: M Time Horizon: T Measurements: K Quantity of interest: user i, item j, measurement k at time t Problem statement: Estimate the above using noisy, sparse observations	$L_{ijk}(t)$		
Recommendation: Model Everything	The model $L_{ijk}(t) = f^k_{\text{obs}}(x_i, y_j) + f^k_{\text{latent}}(u_i(t), v_j(t))$ $u_i(\cdot),\ v_j(\cdot)$ time-series A useful special instance $f^k_{\text{obs}}(x_i, y_j) = \alpha^k x_i + \beta^k y_j + \gamma^k$ $f^k_{\text{latent}}(u_i(t), v_j(t)) = \sum_{\ell=1}^{d} u_{i\ell}(t) v_{j\ell}(t) w_{k\ell}$			
Recommendation: Algorithm	1. Content based learning For each measurement K, learn via supervised learning 2. Obtain difference (not learnt through content)	Step 1. Content based learning For each measurement k, learn via supervised learning f^k_{obs}, that is $(\alpha^k, \beta^k, \gamma^k)$ Step 2. Obtain difference (not learnt through content) $L^{\text{diff}}_{ijk}(t) = L_{ijk}(t) - L^{\text{obs}}_{ijk}$ where $L^{\text{obs}}_{ijk} = f^k_{\text{obs}}(x_i, y_j)$		
Recommendation: Algorithm	3. Build stack Pace matrice across entries, slices of tensor That is, for measurement k, create Page matrix with P rows, T/P x N x M columns			
Recommendation: Algorithm	3. Build stack Pace matrice across entries, slices of tensor That is, for measurement k, create Page matrix with P rows, T/P x N x M columns			
Recommendation: Algorithm	3. Build stack Pace matrice across entries, slices of tensor That is, for measurement k, create Page matrix with P rows, T/P x N x M columns 4. Perform matrix estimation on it	Call it Z^k  Step 4. Perform matrix estimation on it to obtain $\widehat{L}^{\text{diff}}_{ijk}(t)$		
Recommendation: Algorithm	5. Final Estimate Some remarks: We flattened tensor in matrix to estimate to L^diff ijk (t) But, it comes at the cost of increased computation	$\widehat{L}_{ijk}(t) = \widehat{L}^{\text{diff}}_{ijk}(t) + L^{\text{obs}}_{ijk}$ Some remarks: We *flattened* tensor in matrix to estimate to $\widehat{L}^{\text{diff}}_{ijk}(t)$		

Post session summary Recommendation systems				
Description	**What**	**Limitations/Assumptions**	**Example/Formula**	**Reference/Comments**
3 dimensions of recommendation systems	1. **Multiple measurements** - Data for observed preferences 2. **Content or Exogenous features**: Features of users/items 3. **Dynamics** - Time varying aspect			
syntehetic prediction	This is a counterfactual prediction where actual prediction and synthetic prediction which will be predicted if the event did not occur to provide analysis of what is the impact of the event		Basque cuntry is in norther spain. It was affected by terrorism from mid to late 70's and it also experienced a decline in the per-capita GDP income around the same time. The question is whether this decline was due to that terrorism or due to some other factors To answer this question, we need to come up with a counterfactual prediction or synthetic prediction of the per-capita GDP of the Basque country if there was no terrorism	
How do we find these predictions	We will use Matrix estimation to determine the synthetic predictions Idea is similar to what we have learned in collaborative filtering We identify the intervention			
Synthetic control	We will observe pre-intervention data, find the similarity of each donor with the target, and use it to create synthetic post-intervention data. This method is called **synthetic control**			
Applications for matrix technique estimation	Predict scores in an ongoing game Can be used to find highlights or breakthroughs in a game or history when comparing the predictions change and some gap between the actual and predicted score can be observed		Forecasting cricket trajectory in an India vs Australia game from 2011	